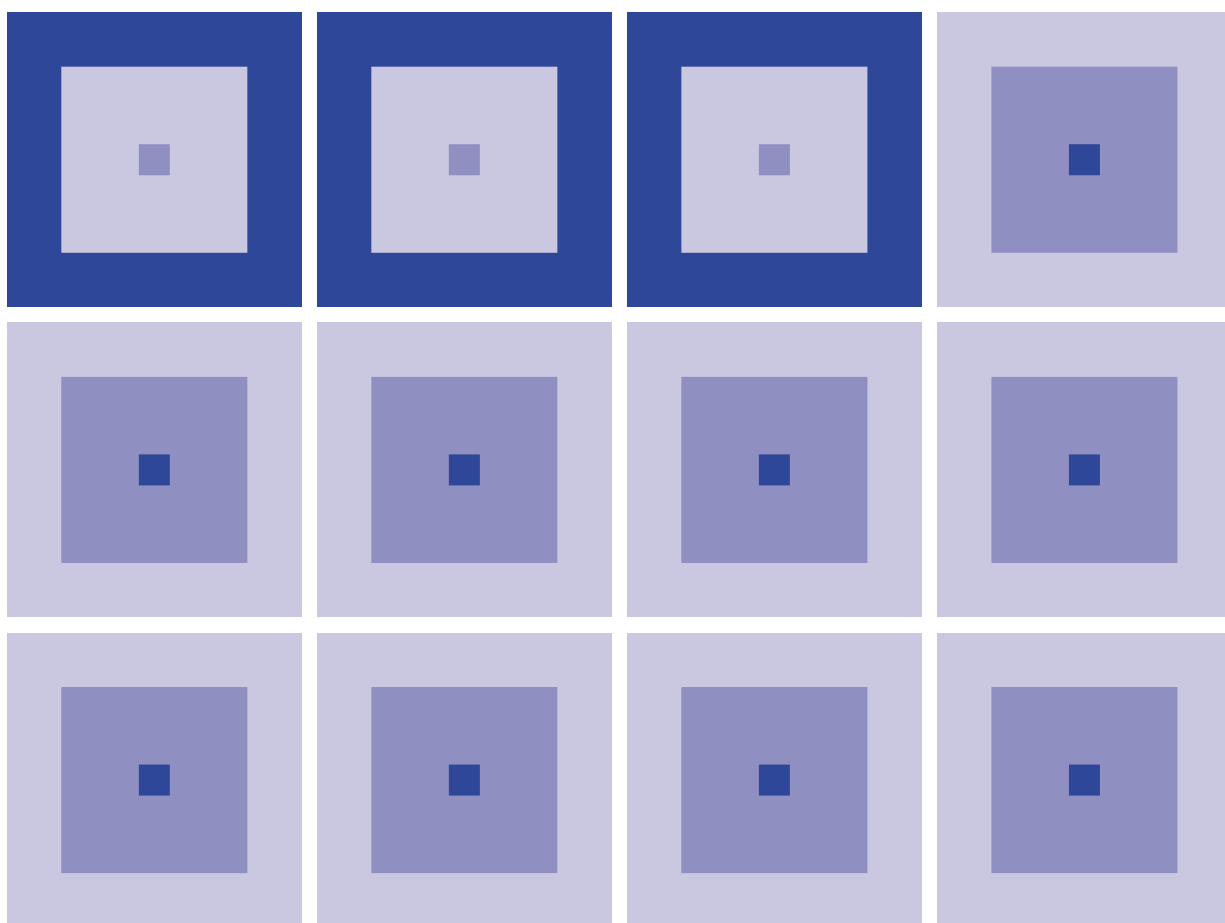


Embedded RAMDAC LCD/CRT Controller

S1D13505F00A

Technical Manual



NOTICE

No part of this material may be reproduced or duplicated in any form or by any means without the written permission of Seiko Epson. Seiko Epson reserves the right to make changes to this material without notice. Seiko Epson does not assume any liability of any kind arising out of any inaccuracies contained in this material or due to its application or use in any product or circuit and, further, there is no representation that this material is applicable to products requiring high level reliability, such as medical products. Moreover, no license to any intellectual property rights is granted by implication or otherwise, and there is no representation or warranty that anything made in accordance with this material will be free from any patent or copyright infringement of a third party. This material or portions thereof may contain technology or the subject relating to strategic products under the control of the Foreign Exchange and Foreign Trade Law of Japan and may require an export license from the Ministry of International Trade and Industry or other approval from another government agency.

MS-DOS and Windows are registered trademarks of Microsoft Corporation, U.S.A.

PC-DOS, PC/AT, VGA, EGA and IBM are registered trademarks of International Business Machines Corporation, U.S.A.

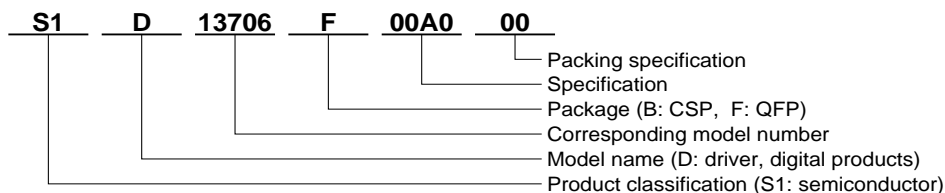
All other product names mentioned herein are trademarks and/or registered trademarks of their respective owners.

The information of the product number change

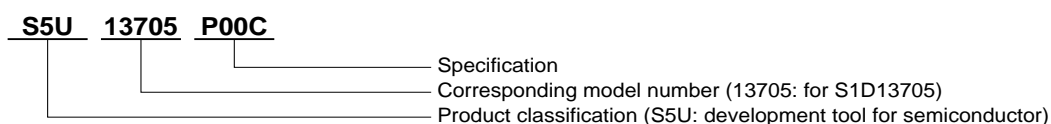
Starting April 1, 2001, the product number will be changed as listed below. To order from April 1, 2001 please use the new product number. For further information, please contact Epson sales representative.

Configuration of product number

● Devices



● Evaluation Board



Comparison table between new and previous number

● S1D13305 Series

Previous No.	New No.
SED1335 Series	S1D13305 Series
SED1335D0A	S1D13305D00A
SED1335F0A	S1D13305F00A
SED1335F0B	S1D13305F00B

● S1D1370x Series

Previous No.	New No.
SED137x Series	S1D1370x Series
SED1374F0A	S1D13704F00A
SED1375F0A	S1D13705F00A
SED1376B0A	S1D13706B00A
SED1376F0A	S1D13706F00A
SED1378 Series	S1D13708 Series

● S1D1380x Series

Previous No.	New No.
SED138x Series	S1D1380x Series
SED1386F0A	S1D13806F00A

● S1D1350x Series

Previous No.	New No.
SED135x Series	S1D1350x Series
SED1353D0A	S1D13503D00A
SED1353F0A	S1D13503F00A
SED1353F1A	S1D13503F01A
SED1354F0A	S1D13504F00A
SED1354F1A	S1D13504F01A
SED1354F2A	S1D13504F02A
SED1355F0A	S1D13505F00A
SED1356F0A	S1D13506F00A

● S1D13A0x Series

Previous No.	New No.
SED13Ax Series	S1D13A0x Series
SED13A3F0A	S1D13A03F00A
SED13A3B0B	S1D13A03B00B
SED13A4B0B	S1D13A04B00B

Comparison table between new and previous number of Evaluation Boards

● S1D1350x Series

Previous No.	New No.
SDU1353#0C	S5U13503P00C
SDU1354#0C	S5U13504P00C
SDU1355#0C	S5U13505P00C
SDU1356#0C	S5U13506P00C

● S1D1370x Series

Previous No.	New No.
SDU1374#0C	S5U13704P00C
SDU1375#0C	S5U13705P00C
SDU1376#0C	S5U13706P00C
SDU1376BVR	S5U13706B32R
SDU1378#0C	S5U13708P00C

● S1D1380x Series

Previous No.	New No.
SDU1386#0C	S5U13806P00C

● S1D13A0x Series

Previous No.	New No.
SDU13A3#0C	S5U13A03P00C
SDU13A4#0C	S5U13A04P00C

S1D13505F00A Technical Manual

HARDWARE FUNCTIONAL SPECIFICATION

PROGRAMMING NOTES AND EXAMPLES

UTILITIES

**S5U13505P00C ISA BUS EVALUATION
BOARD USER'S MANUAL**

APPLICATION NOTES

WINDOWS® CE DISPLAY DRIVERS

S1D13505F00A

Embedded RAMDAC LCD/CRT Controller

Hardware Functional Specification

Table of Contents

1	INTRODUCTION	1-1
1.1	Scope	1-1
1.2	Overview Description	1-1
2	FEATURES	1-2
2.1	Memory Interface	1-2
2.2	CPU Interface	1-2
2.3	Display Support	1-2
2.4	Display Modes	1-3
2.5	Display Features	1-3
2.6	Clock Source	1-3
2.7	Miscellaneous	1-3
3	TYPICAL SYSTEM IMPLEMENTATION DIAGRAMS	1-4
4	INTERNAL DESCRIPTION	1-9
4.1	Block Diagram Showing Datapaths	1-9
4.2	Block Descriptions	1-9
	Register	1-9
	Host Interface	1-9
	CPU R/W	1-9
	Memory Controller	1-9
	Display FIFO	1-9
	Cursor FIFO	1-10
	Look-Up Tables	1-10
	CRTC	1-10
	LCD Interface	1-10
	DAC	1-10
	Power Save	1-10
	Clocks	1-10
5	PINS	1-11
5.1	Pinout Diagram	1-11
5.2	Pin Description	1-12
	Host Interface	1-13
	Memory Interface	1-17
	LCD Interface	1-18
	CRT Interface	1-18
	Miscellaneous	1-19
5.3	Summary of Configuration Options	1-20
5.4	Multiple Function Pin Mapping	1-20
5.5	CRT Interface	1-23
6	D.C. CHARACTERISTICS	1-24
7	A.C. CHARACTERISTICS	1-26
7.1	CPU Interface Timing	1-26
	SH-4 Interface Timing	1-26
	SH-3 Interface Timing	1-28
	MC68K Bus 1 Interface Timing (e.g. MC68000)	1-30
	MC68K Bus 2 Interface Timing (e.g. MC68030)	1-32
	PC Card Interface Timing	1-34
	Generic Interface Timing	1-36
	MIPS/ISA Interface Timing	1-38
	Philips Interface Timing (e.g. PR31500/PR31700)	1-40
	Toshiba Interface Timing (e.g. TX3912)	1-42
	PowerPC Interface Timing (e.g. MPC8xx, MC68040, Coldfire)	1-45

7.2	Clock Input Requirements	1-46
7.3	Memory Interface Timing.....	1-47
	EDO-DRAM Read/Write/Read-Write Timing.....	1-47
	EDO-DRAM CAS Before RAS Refresh Timing.....	1-49
	EDO-DRAM Self-Refresh Timing.....	1-50
	FPM-DRAM Read / Write / Read - Write Timing	1-51
	FPM-DRAM CAS Before RAS Refresh Timing	1-53
	FPM-DRAM Self-Refresh Timing.....	1-54
7.4	Power Sequencing	1-55
	LCD Power Sequencing.....	1-55
	Power Save Status	1-56
7.5	Display Interface.....	1-57
	4-Bit Single Monochrome Passive LCD Panel Timing	1-57
	8-Bit Single Monochrome Passive LCD Panel Timing	1-59
	4-Bit Single Color Passive LCD Panel Timing	1-61
	8-Bit Single Color Passive LCD Panel Timing (Format 1).....	1-63
	8-Bit Single Color Passive LCD Panel Timing (Format 2).....	1-65
	16-Bit Single Color Passive LCD Panel Timing	1-67
	8-Bit Dual Monochrome Passive LCD Panel Timing.....	1-69
	8-Bit Dual Color Passive LCD Panel Timing.....	1-71
	16-Bit Dual Color Passive LCD Panel Timing	1-73
	16-Bit TFT/D-TFD Panel Timing	1-75
	CRT Timing.....	1-77
8	REGISTERS.....	1-79
8.1	Register Mapping	1-79
8.2	Register Descriptions	1-80
	Revision Code Register	1-80
	Memory Configuration Registers	1-80
	Panel/Monitor Configuration Registers	1-81
	Display Configuration Registers.....	1-86
	Clock Configuration Register	1-90
	Power Save Configuration Registers	1-91
	Miscellaneous Registers	1-92
	Look-Up Table Registers	1-97
	Ink/Cursor Registers	1-98
9	DISPLAY BUFFER	1-101
9.1	Image Buffer	1-102
9.2	Ink/Cursor Buffers	1-102
9.3	Half Frame Buffer	1-102
10	DISPLAY CONFIGURATION	1-103
10.1	Display Mode Data Format.....	1-103
10.2	Image Manipulation	1-105
11	LOOK-UP TABLE ARCHITECTURE.....	1-106
11.1	Monochrome Modes.....	1-106
	1 Bit-Per-Pixel Monochrome Mode	1-106
	2 Bit-Per-Pixel Monochrome Mode	1-106
	4 Bit-Per-Pixel Monochrome Mode	1-107
11.2	Color Display Modes	1-108
	1 Bit-Per-Pixel Color Mode	1-108
	2 Bit-Per-Pixel Color Mode.....	1-109
	4 Bit-Per-Pixel Color Mode.....	1-110
	8 Bit-Per-Pixel Color Mode.....	1-111
12	INK/CURSOR ARCHITECTURE	1-113
12.1	Ink/Cursor Buffers	1-113
12.2	Ink/Cursor Data Format	1-113

12.3 Ink/Cursor Image Manipulation	1-114
Ink Image	1-114
Cursor Image	1-114
13 SwiVeLView™	1-115
13.1 Concept	1-115
13.2 Image Manipulation in SwiVeLView™	1-116
13.3 Physical Memory Requirement.....	1-117
13.4 Limitations	1-118
14 CLoCKING	1-119
14.1 Maximum MCLK: PCLK Ratios	1-119
14.2 Frame Rate Calculation.....	1-120
14.3 Bandwidth Calculation	1-122
15 POWER SAVE MODES.....	1-125
16 MECHANICAL DATA	1-126

List of Figures

Figure 3-1	Typical System Diagram (SH-4 Bus, 256Kx16 FPM/EDO-DRAM).....	1-4
Figure 3-2	Typical System Diagram (SH-3 Bus, 256Kx16 FPM/EDO-DRAM).....	1-4
Figure 3-3	Typical System Diagram (MC68K Bus 1, 16-Bit 68000, 256Kx16 FPM/EDO-DRAM).....	1-5
Figure 3-4	Typical System Diagram (MC68K Bus 2, 32-Bit 68030, 256Kx16 FPM/EDO-DRAM).....	1-5
Figure 3-5	Typical System Diagram (Generic Bus, 1Mx16 FPM/EDO-DRAM).....	1-6
Figure 3-6	Typical System Diagram (NEC VR41xx (MIPS) Bus, 1Mx16 FPM/EDO-DRAM).....	1-6
Figure 3-7	Typical System Diagram (Philips PR31500/PR31700 Bus, 1Mx16 FPM/EDO-DRAM).....	1-7
Figure 3-8	Typical System Diagram (Toshiba TX3912 Bus, 1Mx16 FPM/EDO-DRAM).....	1-7
Figure 3-9	Typical System Diagram (Power PC Bus, 256Kx16 FPM/EDO-DRAM).....	1-8
Figure 3-10	Typical System Diagram (PC Card (PCMCIA) Bus, 1Mx16 FPM/EDO-DRAM).....	1-8
Figure 4-1	System Block Diagram Showing Datapaths	1-9
Figure 5-1	Pinout Diagram	1-11
Figure 5-2	External Circuitry for CRT Interface	1-23
Figure 7-1	SH-4 Timing	1-26
Figure 7-2	SH-3 Timing	1-28
Figure 7-3	MC68000 Timing.....	1-30
Figure 7-4	MC68030 Timing.....	1-32
Figure 7-5	PC Card Interface Timing	1-34
Figure 7-6	Generic Timing.....	1-36
Figure 7-7	MIPS/ISA Timing.....	1-38
Figure 7-8	Philips Timing.....	1-40
Figure 7-9	Clock Input Requirements for BUSCLK Using Philips Local Bus	1-41
Figure 7-10	Toshiba Timing	1-42
Figure 7-11	Clock Input Requirements.....	1-44
Figure 7-12	PowerPC Timing	1-45
Figure 7-13	Clock Input Requirements.....	1-46
Figure 7-14	EDO-DRAM Read/Write Timing.....	1-47
Figure 7-15	EDO-DRAM Read-Write Timing.....	1-47
Figure 7-16	EDO-DRAM CAS Before RAS Refresh Write Timing	1-49
Figure 7-17	EDO-DRAM Self-Refresh Timing.....	1-50
Figure 7-18	FPM-DRAM Read/Write Timing.....	1-51
Figure 7-19	FPM-DRAM Read-Write Timing.....	1-51
Figure 7-20	FPM-DRAM CAS before RAS Refresh Timing	1-53
Figure 7-21	FPM-DRAM Self-Refresh Timing.....	1-54
Figure 7-22	LCD Panel Power Off / Power On Timing. Drawn with LCDPWR Set to Active High Polarity1-55	
Figure 7-23	Power Save Status and Local Bus Memory Access Relative to Power Save Mode.....	1-56
Figure 7-24	4-Bit Single Monochrome Passive LCD Panel Timing.....	1-57
Figure 7-25	4-Bit Single Monochrome Passive LCD Panel A.C. Timing.....	1-58
Figure 7-26	8-Bit Single Monochrome Passive LCD Panel Timing.....	1-59
Figure 7-27	8-Bit Single Monochrome Passive LCD Panel A.C. Timing.....	1-60
Figure 7-28	4-Bit Single Color Passive LCD Panel Timing	1-61
Figure 7-29	4-Bit Single Color Passive LCD Panel A.C.Timing	1-62
Figure 7-30	8-Bit Single Color Passive LCD Panel Timing (Format 1).....	1-63
Figure 7-31	8-Bit Single Color Passive LCD Panel A.C. Timing (Format 1).....	1-64
Figure 7-32	8-Bit Single Color Passive LCD Panel Timing (Format 2).....	1-65
Figure 7-33	8-Bit Single Color Passive LCD Panel A.C. Timing (Format 2).....	1-66
Figure 7-34	16-Bit Single Color Passive LCD Panel Timing	1-67
Figure 7-35	16-Bit Single Color Passive LCD Panel A.C. Timing	1-68

Figure 7-36	8-Bit Dual Monochrome Passive LCD Panel Timing	1-69
Figure 7-37	8-Bit Dual Monochrome Passive LCD Panel A.C. Timing	1-70
Figure 7-38	8-Bit Dual Color Passive LCD Panel Timing	1-71
Figure 7-39	8-Bit Dual Color Passive LCD Panel A.C. Timing	1-72
Figure 7-40	16-Bit Dual Color Passive LCD Panel Timing	1-73
Figure 7-41	16-Bit Dual Color Passive LCD Panel A.C. Timing	1-74
Figure 7-42	16-Bit TFT/D-TFD Panel Timing	1-75
Figure 7-43	16-Bit TFT/D-TFD A.C. Timing	1-76
Figure 7-44	CRT Timing	1-77
Figure 7-45	CRT A.C. Timing	1-78
Figure 9-1	Display Buffer Addressing	1-101
Figure 10-1	1/2/4/8 Bit-Per-Pixel Format Memory Organization	1-103
Figure 10-2	15/16 Bit-Per-Pixel Format Memory Organization	1-104
Figure 10-3	Image Manipulation	1-105
Figure 11-1	1 Bit-per-pixel Monochrome Mode Data Output Path	1-106
Figure 11-2	2 Bit-per-pixel Monochrome Mode Data Output Path	1-106
Figure 11-3	4 Bit-per-pixel Monochrome Mode Data Output Path	1-107
Figure 11-4	1 Bit-per-pixel Color Mode Data Output Path	1-108
Figure 11-5	2 Bit-per-pixel Color Mode Data Output Path	1-109
Figure 11-6	4 Bit-per-pixel Color Mode Data Output Path	1-110
Figure 11-7	8 Bit-per-pixel Color Mode Data Output Path	1-111
Figure 12-1	Ink/Cursor Data Format	1-113
Figure 12-2	Cursor Positioning	1-114
Figure 13-1	Relationship Between the Screen Image and the Image Residing in the Display Buffer	1-115
Figure 16-1	Mechanical Drawing QFP15	1-126

List of Tables

Table 5-1	Host Interface Pin Descriptions.....	1-13
Table 5-2	Memory Interface Pin Descriptions	1-17
Table 5-3	LCD Interface Pin Descriptions	1-18
Table 5-4	Clock Input Pin Description	1-18
Table 5-5	Miscellaneous Interface Pin Descriptions	1-19
Table 5-6	Summary of Power On / Reset Options.....	1-20
Table 5-7	CPU Interface Pin Mapping	1-20
Table 5-8	Memory Interface Pin Mapping	1-21
Table 5-9	LCD Interface Pin Mapping.....	1-22
Table 6-1	Absolute Maximum Ratings	1-24
Table 6-2	Recommended Operating Conditions	1-24
Table 6-3	Electrical Characteristics for $V_{DD} = 5.0V$ Typical	1-24
Table 6-4	Electrical Characteristics for $V_{DD} = 3.3V$ Typical	1-24
Table 6-5	Electrical Characteristics for $V_{DD} = 3.0V$ Typical	1-25
Table 7-1	SH-4 Timing	1-27
Table 7-2	SH-3 Timing	1-29
Table 7-3	MC68000 Timing.....	1-31
Table 7-4	MC68030 Timing.....	1-33
Table 7-5	PC Card Interface Timing	1-35
Table 7-6	Generic Timing.....	1-37
Table 7-7	MIPS/ISA Timing.....	1-39
Table 7-8	Philips Timing.....	1-41
Table 7-9	Clock Input Requirements for BUSCLK Using Philips Local Bus	1-41
Table 7-10	Toshiba Timing	1-43
Table 7-11	Clock Input Requirements for BUSCLK Using Toshiba Local Bus	1-44
Table 7-12	PowerPC Timing	1-45
Table 7-13	Clock Input Requirements for CLKI Divided Down Internally ($MCLK = CLKI/2$)	1-46
Table 7-14	Clock Input Requirements for CLKI	1-46
Table 7-15	EDO DRAM Read Timing	1-48
Table 7-16	EDO DRAM CAS Before RAS Refresh Write Timing.....	1-49
Table 7-17	EDO-DRAM Self-Refresh Timing.....	1-50
Table 7-18	FPM-DRAM Read/Write/Read-Write Timing.....	1-52
Table 7-19	FPM-DRAM CAS before RAS Refresh Timing	1-53
Table 7-20	FPM DRAM Self-Refresh Timing	1-54
Table 7-21	LCD Panel Power Off/ Power On.....	1-55
Table 7-22	Power Save Status and Local Bus Memory Access Relative to Power Save Mode.....	1-56
Table 7-23	4-Bit Single Monochrome Passive LCD Panel A.C. Timing.....	1-58
Table 7-24	8-Bit Single Monochrome Passive LCD Panel A.C. Timing.....	1-60
Table 7-25	4-Bit Single Color Passive LCD Panel A.C.Timing	1-62
Table 7-26	8-Bit Single Color Passive LCD Panel A.C. Timing (Format 1).....	1-64
Table 7-27	8-Bit Single Color Passive LCD Panel A.C. Timing (Format 2).....	1-66
Table 7-28	16-Bit Single Color Passive LCD Panel A.C. Timing	1-68
Table 7-29	8-Bit Dual Monochrome Passive LCD Panel A.C. Timing.....	1-70
Table 7-30	8-Bit Dual Color Passive LCD Panel A.C. Timing	1-72
Table 7-31	16-Bit Dual Color Passive LCD Panel A.C. Timing.....	1-74
Table 7-32	16-Bit TFT/D-TFD A.C. Timing.....	1-76
Table 7-33	CRT A.C. Timing.....	1-78
Table 8-1	S1D13505 Addressing	1-79
Table 8-2	DRAM Refresh Rate Selection	1-80
Table 8-3	Panel Data Width Selection	1-81
Table 8-4	FPLINE Polarity Selection.....	1-83
Table 8-5	FPFRAME Polarity Selection	1-85
Table 8-6	Simultaneous Display Option Selection	1-86

Table 8-7	Bits-Per-Pixel Selection	1-87
Table 8-8	Pixel Panning Selection	1-89
Table 8-9	PCLK Divide Selection	1-90
Table 8-10	Suspend Refresh Selection	1-91
Table 8-11	MA/GPIO Pin Functionality	1-93
Table 8-12	Minimum Memory Timing Selection	1-95
Table 8-13	RAS#-to-CAS# Delay Timing Select	1-95
Table 8-14	RAS# Precharge Timing Select	1-95
Table 8-15	Optimal NRC, NRP, and NRCD Values at Maximum MCLK Frequency	1-96
Table 8-16	Minimum Memory Timing Selection	1-96
Table 8-17	Ink/Cursor Selection	1-98
Table 8-18	Ink/Cursor Start Address Encoding	1-100
Table 8-19	Recommended Alternate FRM Scheme	1-100
Table 9-1	S1D13505 Addressing	1-101
Table 12-1	Ink/Cursor Data Format	1-113
Table 12-2	Ink/Cursor Color Select	1-113
Table 13-1	Minimum DRAM Size Required for SwivleView™	1-118
Table 14-1	Maximum PCLK Frequency with EDO-DRAM	1-119
Table 14-2	Maximum PCLK Frequency with FPM-DRAM	1-120
Table 14-3	Example Frame Rates with Ink Disabled	1-121
Table 14-4	Number of MCLKs Required for Various Memory Access	1-122
Table 14-5	Total # MCLKs Taken for Display Refresh	1-123
Table 14-6	Theoretical Maximum Bandwidth M byte/sec, Cursor/Ink Disabled	1-124
Table 15-1	Power Save Mode Function Summary	1-125
Table 15-2	Pin States in Power-Save Modes	1-125

1 INTRODUCTION

1.1 Scope

This is the Hardware Functional Specification for the S1D13505 Embedded RAMDAC LCD/CRT Controller. Included in this document are timing diagrams, AC and DC characteristics, register descriptions, and power management descriptions. This document is intended for two audiences: Video Subsystem Designers and Software Developers.

1.2 Overview Description

The S1D13505 is a color/monochrome LCD/CRT graphics controller interfacing to a wide range of CPUs and display devices. The S1D13505 architecture is designed to meet the low cost, low power requirements of the embedded markets, such as Mobile Communications, Hand-Held PCs, and Office Automation.

The S1D13505 supports multiple CPUs, all LCD panel types, CRT, and additionally provides a number of differentiating features. Products requiring a “Portrait” mode display can take advantage of the SwivelView™ feature. Simultaneous, Virtual and Split Screen Display are just some of the display modes supported, while the Hardware Cursor, Ink Layer, and the Memory Enhancement Registers offer substantial performance benefits. These features, combined with the S1D13505’s Operating System independence, make it an ideal display solution for a wide variety of applications.

2 *FEATURES*

2.1 *Memory Interface*

- 16-bit DRAM interface:
 - EDO-DRAM up to 40MHz data rate (80M bytes per second).
 - FPM-DRAM up to 25MHz data rate (50M bytes per second).
- Memory size options:
 - 512K bytes using one 256K×16 device.
 - 2M bytes using one 1M×16 device.
- Performance Enhancement Register to tailor the memory control output timing for the DRAM device.

2.2 *CPU Interface*

- Supports the following interfaces:
 - 8/16-bit SH-4 bus interface.
 - 8/16-bit SH-3 bus interface.
 - 8/16-bit interface to 8/16/32-bit MC68000 microprocessors/microcontrollers.
 - 8/16-bit interface to 8/16/32-bit MC68030 microprocessors/microcontrollers.
 - Philips PR31500/PR31700 (MIPS).
 - Toshiba TX3912 (MIPS).
 - Philips PR31500/PR31700.
 - 16-bit Power PC (MPC821) microprocessor.
 - 16-bit Epson E0C33 microprocessor.
 - PC Card (PCMCIA).
 - StrongARM (PC Card).
 - NEC VR41xx (MIPS).
 - ISA bus.
- Supports the following interface with external logic:
 - GX486 microprocessor.
- One-stage write buffer for minimum wait-state CPU writes.
- Registers are memory-mapped – the M/R# pin selects between the display buffer and register address space.
- The complete 2M byte display buffer address space is addressable as a single linear address space through the 21-bit address bus.

2.3 *Display Support*

- 4/8-bit monochrome passive LCD interface.
- 4/8/16-bit color passive LCD interface.
- Single-panel, single-drive displays.
- Dual-panel, dual-drive displays.
- Direct support for 9/12-bit TFT/D-TFD; 18-bit TFT/D-TFD is supported up to 64K color depth (16-bit data).
- Embedded RAMDAC (DAC) with direct analog CRT drive.
- Simultaneous display of CRT and passive or TFT/D-TFD panels.

2.4 Display Modes

- 1/2/4/8/15/16 bit-per-pixel (bpp) support on LCD/CRT.
- Up to 16 shades of gray using FRM on monochrome passive LCD panels.
- Up to 4096 colors on passive LCD panels; three 256x4 Look-Up Tables (LUT) are used to map 1/2/4/8 bpp modes into these colors, 15/16 bpp modes are mapped directly using the 4 most significant bits of the red, green and blue colors.
- Up to 64K colors on TFT/D-TFD LCD panels and CRT; three 256x4 Look-Up Tables are used to map 1/2/4/8 bpp modes into 4096 colors, 15/16 bpp modes are mapped directly.

2.5 Display Features

- SwivelView™: direct hardware 90° rotation of display image for “portrait” mode display.
- Split Screen Display: allows two different images to be simultaneously viewed on the same display.
- Virtual Display Support: displays images larger than the display size through the use of panning.
- Double Buffering/multi-pages: provides smooth animation and instantaneous screen update.
- Acceleration of screen updates by allocating full display memory bandwidth to CPU (see REG[23h] bit 7).
- Hardware 64x64 pixel 2-bit cursor or full screen 2-bit ink layer.
- Simultaneous display of CRT and passive panel or TFT/D-TFD panel.
- Normal mode for cases where LCD and CRT screen sizes are identical.
- Line-doubling for simultaneous display of 240-line images on 240-line LCD and 480-line CRT.
- Even-scan or interlace modes for simultaneous display of 480-line images on 240-line LCD and 480-line CRT.

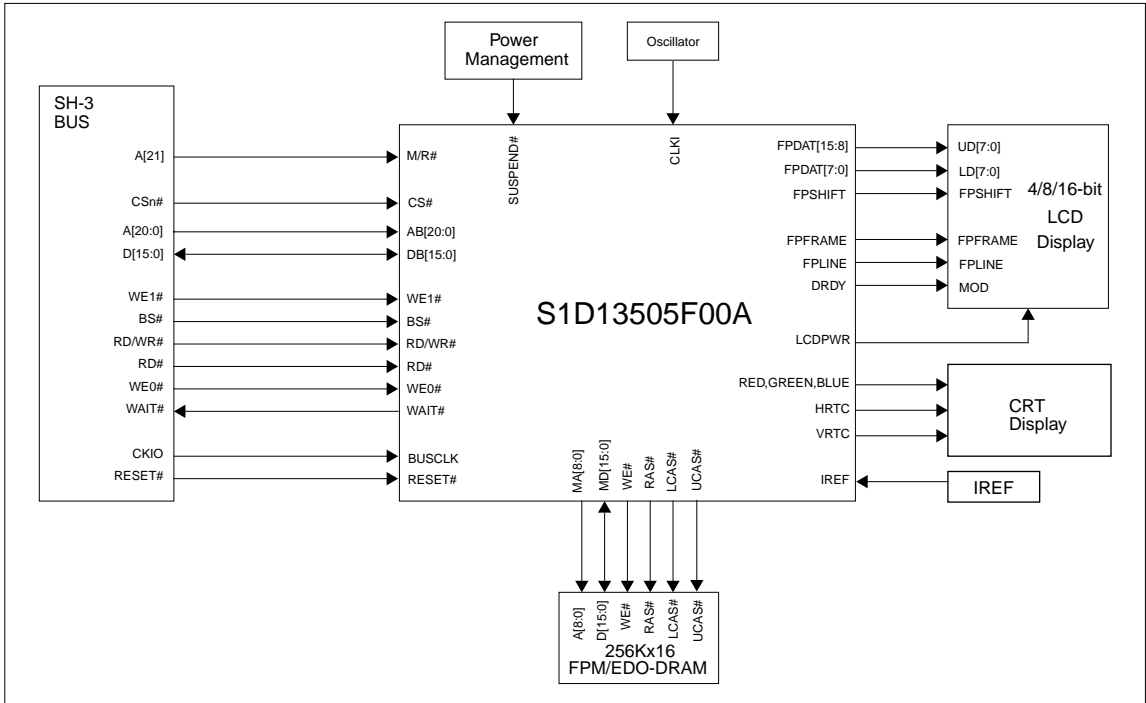
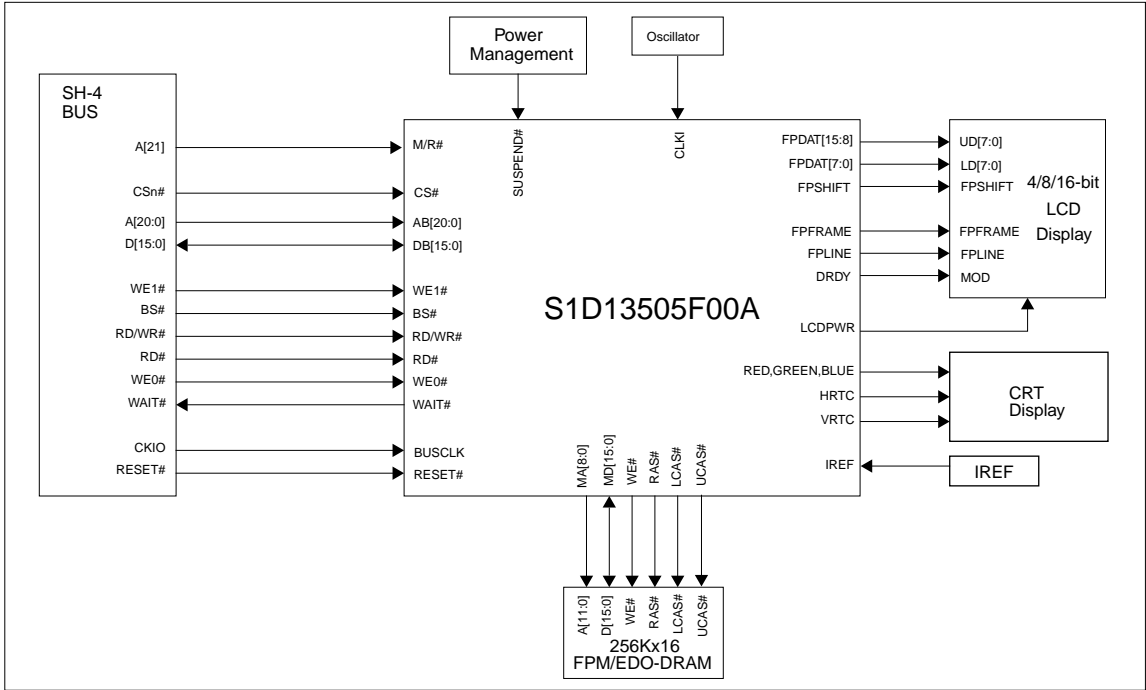
2.6 Clock Source

- Single clock input for both the pixel and memory clocks.
- Memory clock can be input clock or (input clock/2), providing flexibility to use CPU bus clock as input.
- Pixel clock can be the memory clock, (memory clock/2), (memory clock/3) or (memory clock/4).

2.7 Miscellaneous

- The memory data bus, MD[15:0], is used to configure the chip at power-on.
- Three General Purpose Input/Output pins, GPIO[3:1], are available if the upper Memory Address pins are not required for asymmetric DRAM support.
- Suspend power save mode can be initiated by either hardware or software.
- The SUSPEND# pin is used either as an input to initiate Suspend mode, or as a General Purpose Output that can be used to control the LCD backlight. Power-on polarity is selected by an MD configuration pin.
- Operating voltages from 2.7 volts to 5.5 volts are supported
- 128-pin QFP15 surface mount package

3 TYPICAL SYSTEM IMPLEMENTATION DIAGRAMS



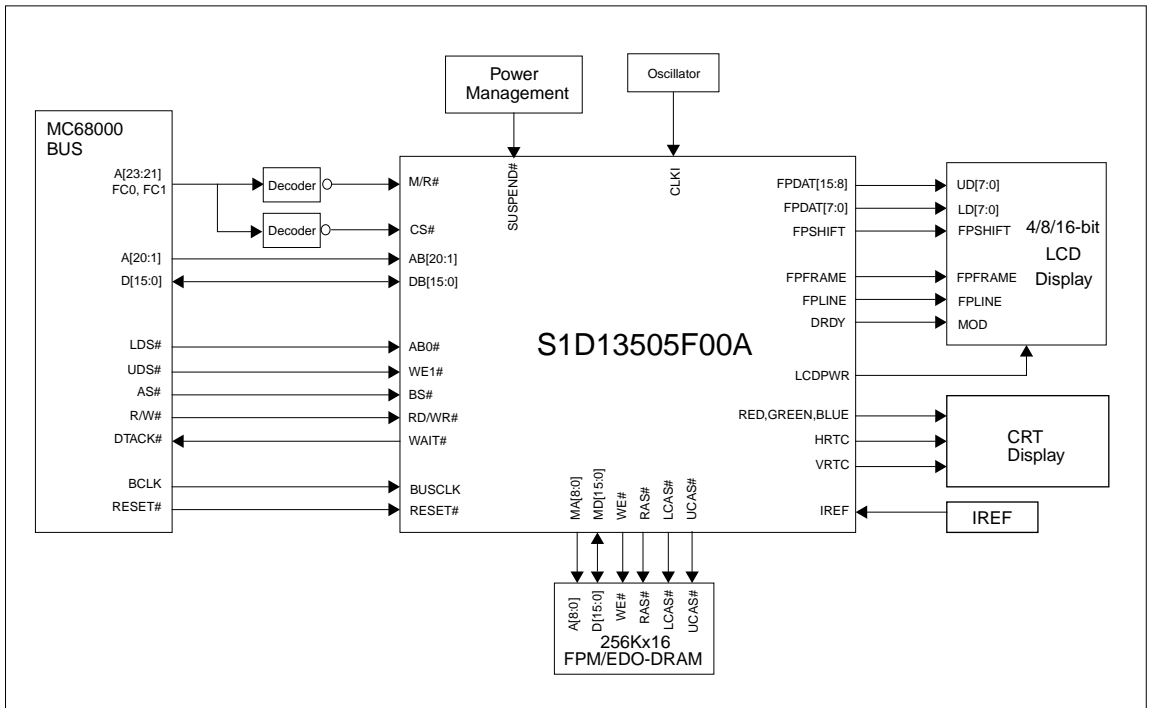


Figure 3-3 Typical System Diagram (MC68K Bus 1, 16-Bit 68000, 256Kx16 FPM/EDO-DRAM)

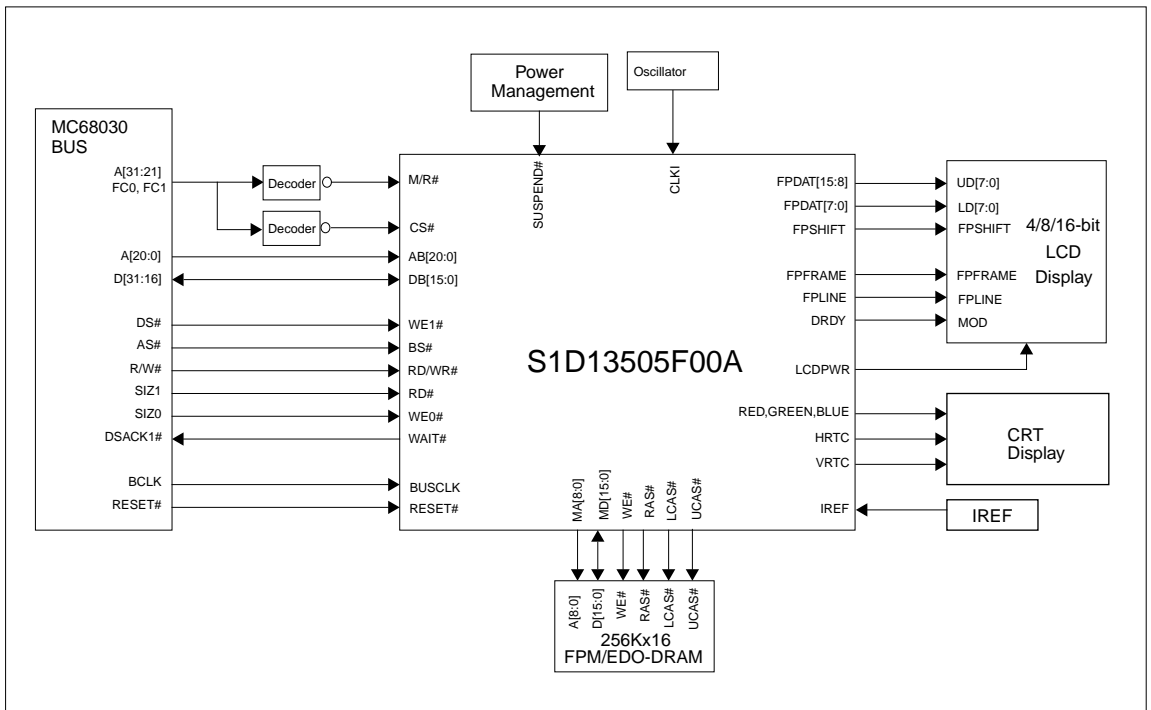


Figure 3-4 Typical System Diagram (MC68K Bus 2, 32-Bit 68030, 256Kx16 FPM/EDO-DRAM)

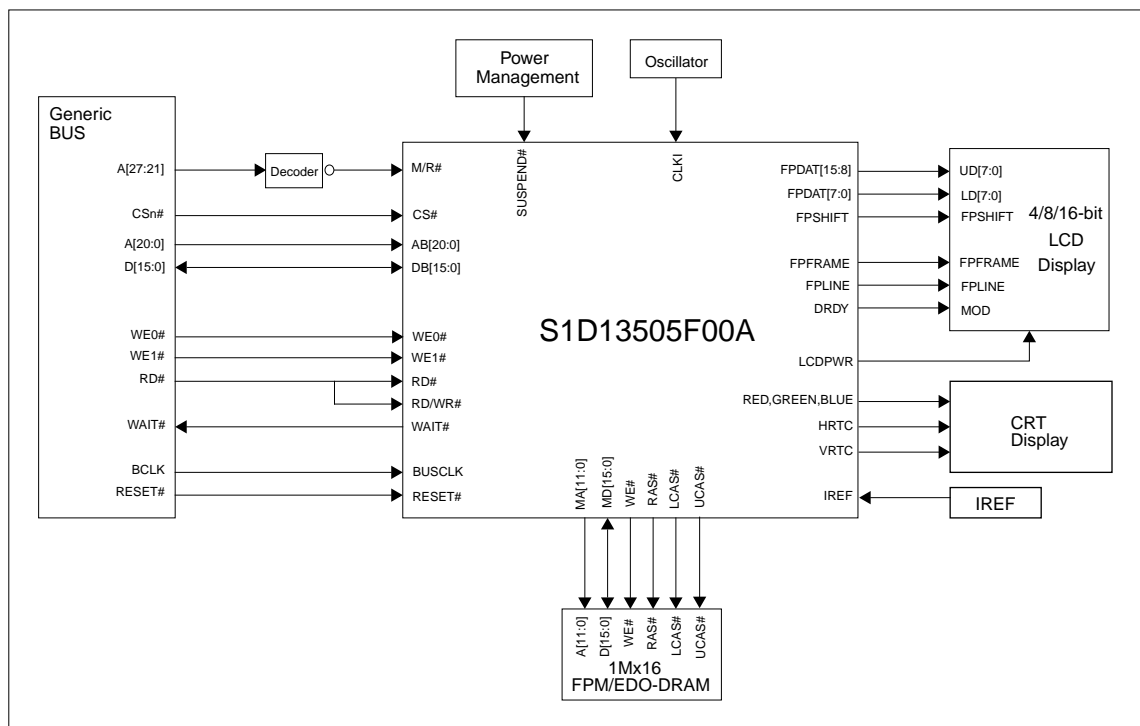


Figure 3-5 Typical System Diagram (Generic Bus, 1Mx16 FPM/EDO-DRAM)

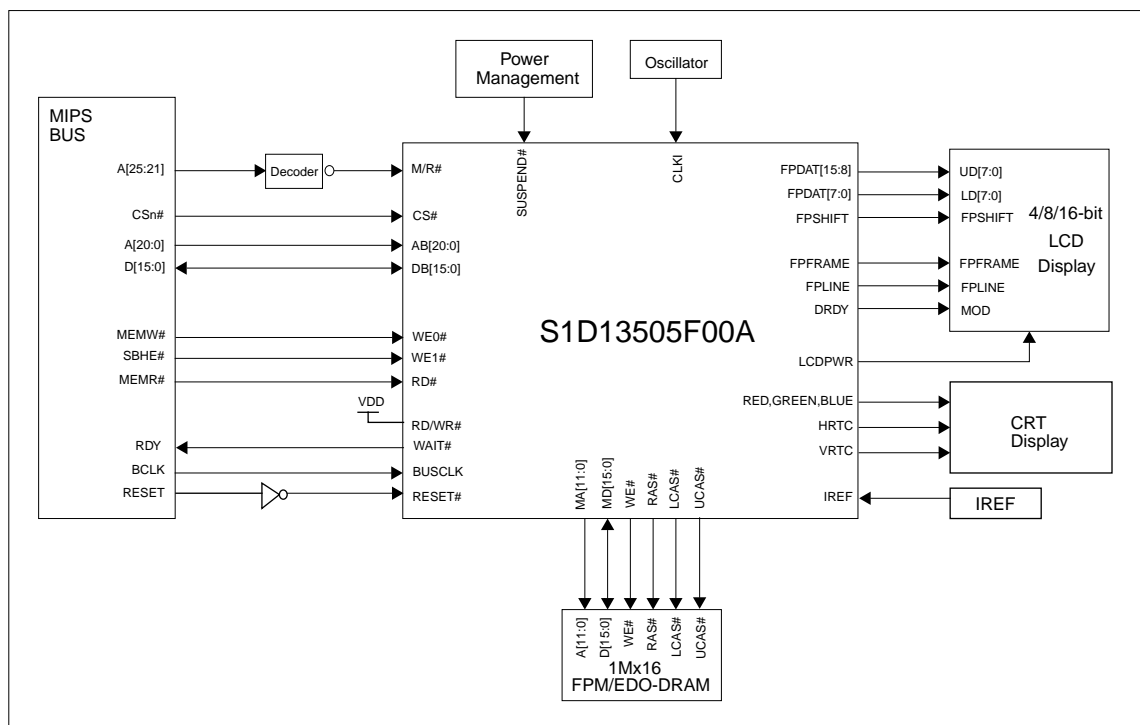


Figure 3-6 Typical System Diagram (NEC Vr41xx (MIPS) Bus, 1Mx16 FPM/EDO-DRAM)

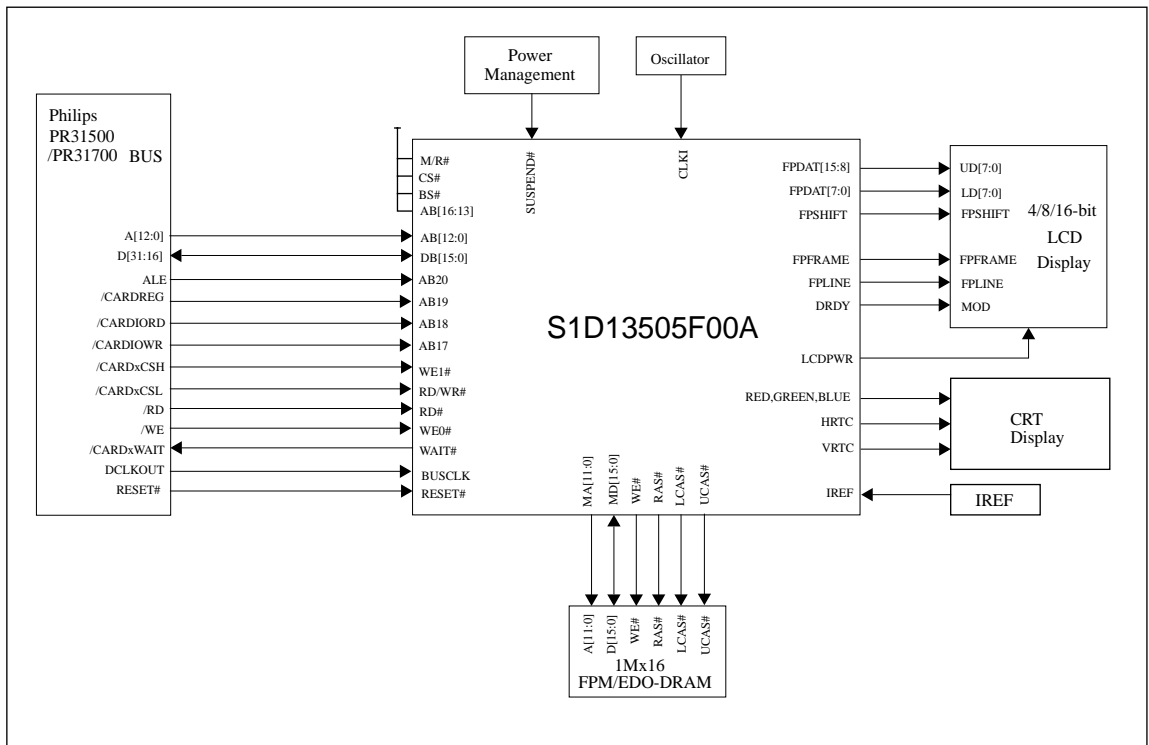


Figure 3-7 Typical System Diagram (Philips PR31500/PR31700 Bus, 1Mx16 FPM/EDO-DRAM)

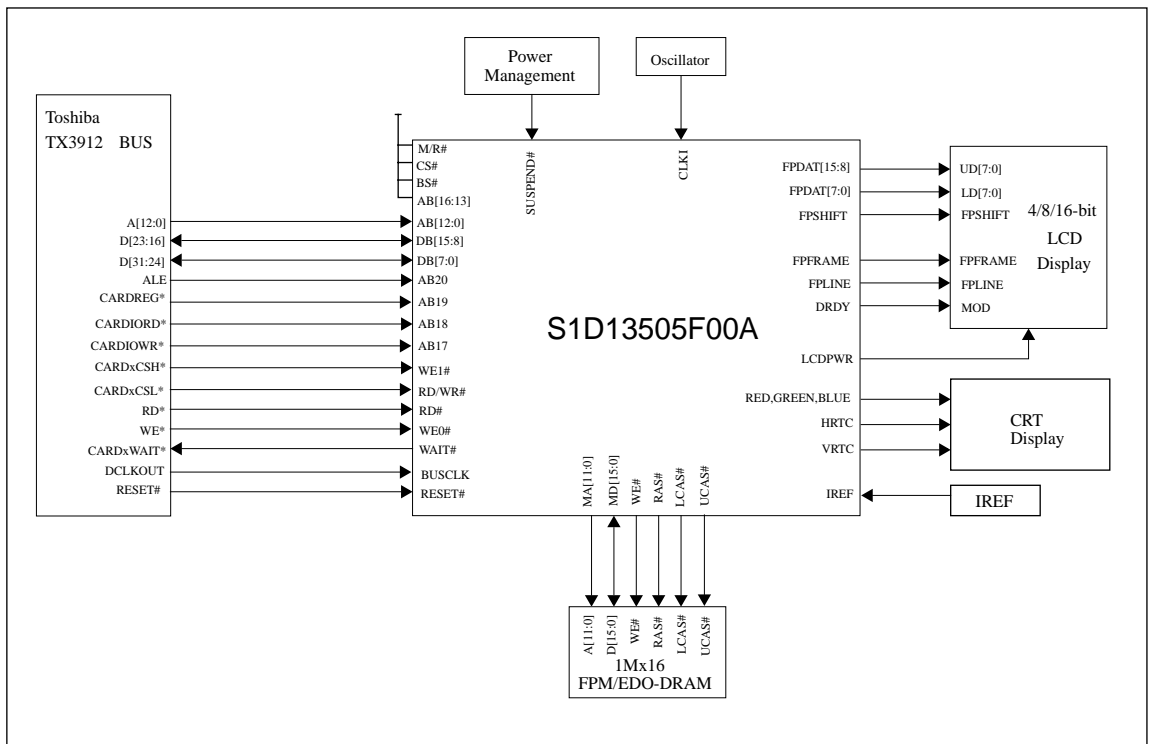


Figure 3-8 Typical System Diagram (Toshiba TX3912 Bus, 1Mx16 FPM/EDO-DRAM)

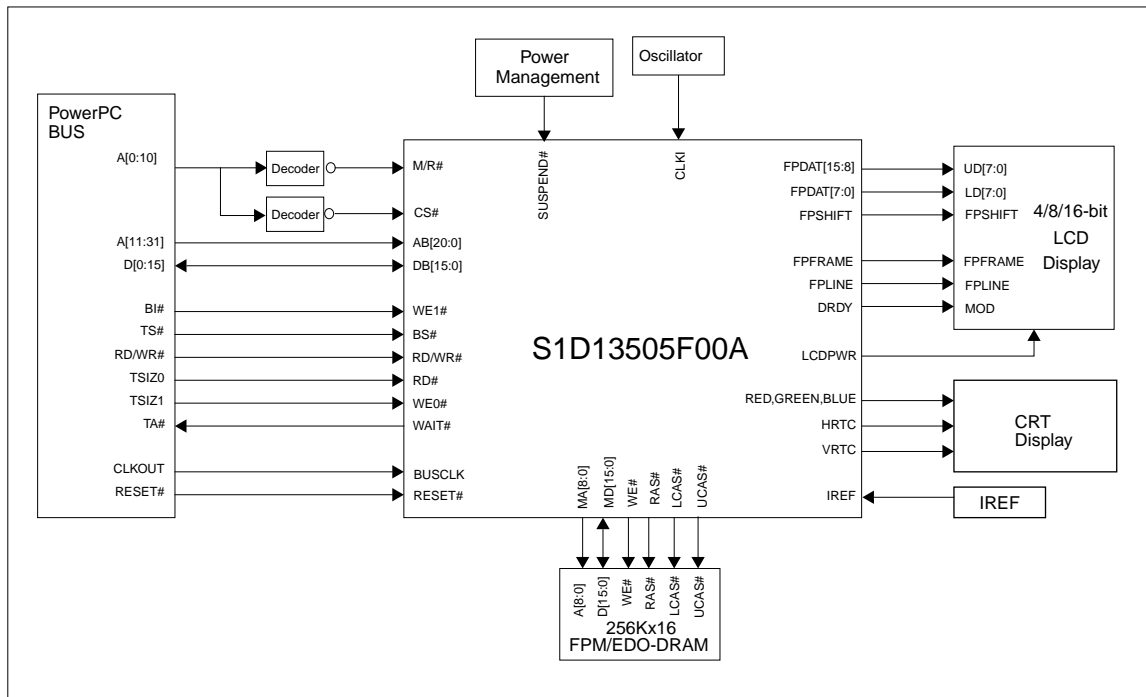


Figure 3-9 Typical System Diagram (Power PC Bus, 256Kx16 FPM/EDO-DRAM)

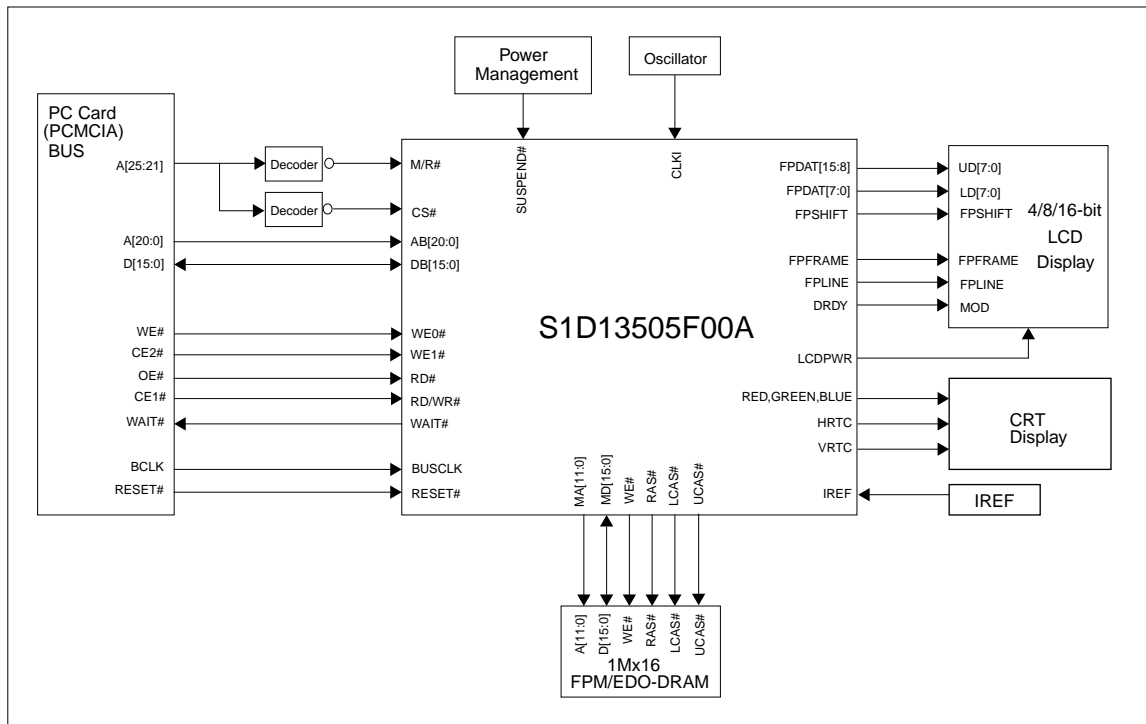


Figure 3-10 Typical System Diagram (PC Card (PCMCIA) Bus, 1Mx16 FPM/EDO-DRAM)

4 INTERNAL DESCRIPTION

4.1 Block Diagram Showing Datapaths

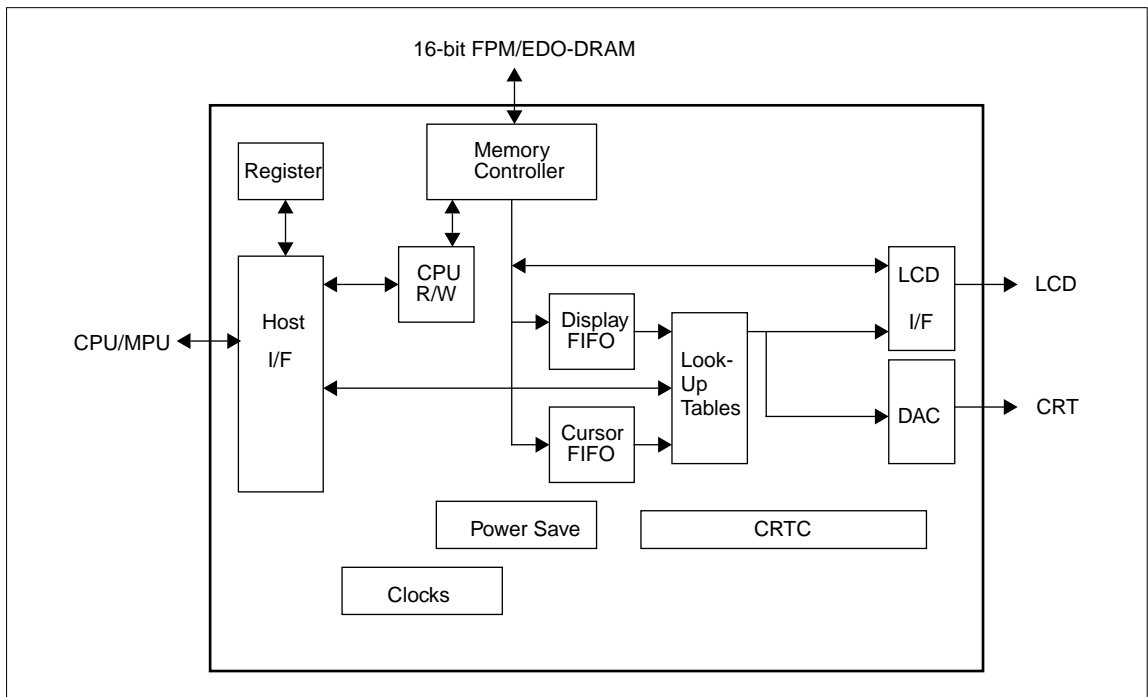


Figure 4-1 System Block Diagram Showing Datapaths

4.2 Block Descriptions

Register

The Register block contains all the register latches.

Host Interface

The Host Interface (I/F) block provides the means for the CPU/MPU to communicate with the display buffer and internal registers via one of the supported bus interfaces.

CPU R/W

The CPU R/W block synchronizes the CPU requests for display buffer access. If SwivelView™ is enabled, the data is rotated in this block.

Memory Controller

The Memory Controller block arbitrates between CPU accesses and display refresh accesses as well as generates the necessary signals to interface to one of the supported 16-bit memory devices (FPM-DRAM or EDO-DRAM).

Display FIFO

The Display FIFO block fetches display data from the Memory Controller for display refresh.

Cursor FIFO

The Cursor FIFO block fetches Cursor/ink data from the Memory Controller for display refresh.

Look-Up Tables

The Look-Up Tables block contains three 256x4 Look-Up Tables (LUT), one for each primary color. In monochrome mode, only the green LUT is selected and used. This block contains anti-sparkle circuitry. The cursor/ink and display data are merged in this block.

CRTC

The CRTC generates the sync timing for the LCD and CRT, defining the vertical and horizontal display periods.

LCD Interface

The LCD Interface block performs Frame Rate Modulation (FRM) for passive LCD panels and generates the correct data format and timing control signals for various LCD and TFT/D-TFD panels.

DAC

The DAC is the Digital to Analog converter for analog CRT support.

Power Save

The Power Save block contains the power save mode circuitry.

Clocks

The Clocks module is the source of all clocks in the chip.

5 PINS

5.1 Pinout Diagram

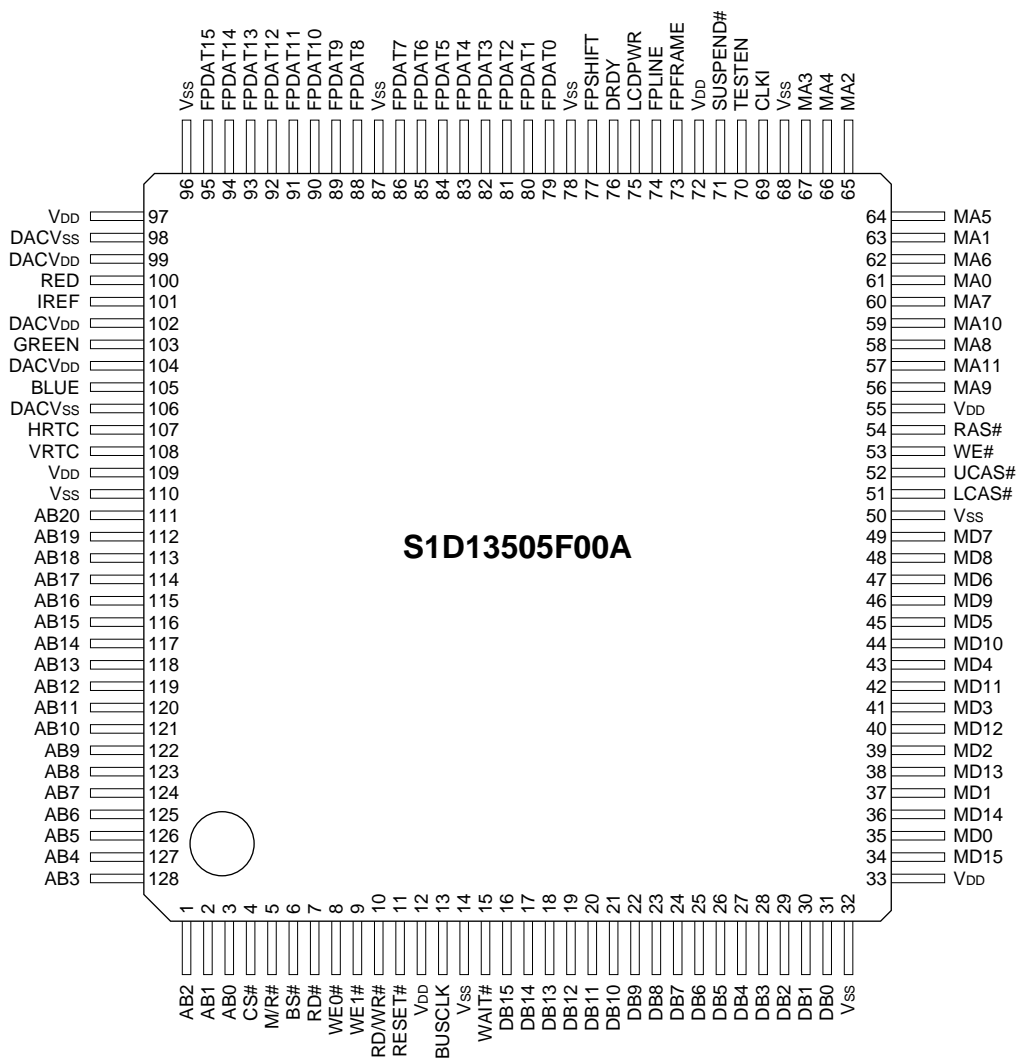


Figure 5-1 Pinout Diagram

128-pin QFP15 surface mount package

5.2 *Pin Description*

Key:

- I** = Input
- O** = Output
- I/O** = Bi-Directional (Input/Output)
- A** = Analog
- P** = Power pin
- C** = CMOS level input
- CD** = CMOS level input with pull down resistor (typical values of 100K Ω /180K Ω at 5V/3.3V respectively)
- CS** = CMOS level Schmitt input
- COx** = CMOS output driver, x denotes driver type (see tables 6-3, 6-4, 6-5 for details)
- TSx** = Tri-state CMOS output driver, x denotes driver type (see tables 6-3, 6-4, 6-5 for details)
- TSxD** = Tri-state CMOS output driver with pull down resistor (typical values of 100K Ω /180K Ω at 5V/3.3V respectively), x denotes driver type (see tables 6-3, 6-4, 6-5 for details)
- CNx** = CMOS low-noise output driver, x denotes driver type (see tables 6-3, 6-4, 6-5 for details)

Host Interface

Table 5-1 Host Interface Pin Descriptions

Pin Name	Type	Pin #	Driver	Reset# State	Description
AB0	I	3	CS	Hi-Z	<ul style="list-style-type: none"> For SH-3/SH-4 Bus, this pin inputs system address bit 0 (A0). For MC68K Bus 1, this pin inputs the lower data strobe (LDS#). For MC68K Bus 2, this pin inputs system address bit 0 (A0). For Generic Bus, this pin inputs system address bit 0 (A0). For MIPS/ISA Bus, this pin inputs system address bit 0 (SA0). For Philips PR31500/31700 Bus, this pin inputs system address bit 0 (A0). For Toshiba TX3912 Bus, this pin inputs system address bit 0 (A0). For PowerPC Bus, this pin inputs system address bit 31 (A31). For PC Card (PCMCIA) Bus, this pin inputs system address bit 0 (A0). See "Table 5-7 CPU Interface Pin Mapping" for summary. See the respective AC Timing diagram for detailed functionality.
AB[12:1]	I	119–128, 1, 2	C	Hi-Z	<ul style="list-style-type: none"> For PowerPC Bus, these pins input the system address bits 19 through 30 (A[19:30]). For all other busses, these pins input the system address bits 12 through 1 (A[12:1]). See "Table 5-7 CPU Interface Pin Mapping" for summary. See the respective AC Timing diagram for detailed functionality.
AB[16:13]	I	115–118	C	Hi-Z	<ul style="list-style-type: none"> For Philips PR31500/31700 Bus, these pins are connected to VDD. For Toshiba TX3912 Bus, these pins are connected to VDD. For PowerPC Bus, these pins input the system address bits 15 through 18 (A[15:18]). For all other busses, these pins input the system address bits 16 through 13 (A[16:13]). See "Table 5-7 CPU Interface Pin Mapping" for summary. See the respective AC Timing diagram for detailed functionality.
AB17#	I	114	C	Hi-Z	<ul style="list-style-type: none"> For Philips PR31500/31700 Bus, this pin inputs the I/O write command (/CARDIOWR). For Toshiba TX3912 Bus, this pin inputs the I/O write command (CARDIOWR*). For PowerPC Bus, this pin inputs the system address bit 14 (A14). For all other busses, this pin inputs the system address bit 17 (A17). See "Table 5-7 CPU Interface Pin Mapping" for summary. See the respective AC Timing diagram for detailed functionality.
AB18	I	113	C	Hi-Z	<ul style="list-style-type: none"> For Philips PR31500/31700 Bus, this pin inputs the I/O read command (/CARDIORD). For Toshiba TX3912 Bus, this pin inputs the I/O read command (CARDIORD*). For PowerPC Bus, this pin inputs the system address bit 13 (A13). For all other busses, this pin inputs the system address bit 18 (A18). See "Table 5-7 CPU Interface Pin Mapping" for summary. See the respective AC Timing diagram for detailed functionality.
AB19	I	112	C	Hi-Z	<ul style="list-style-type: none"> For Philips PR31500/31700 Bus, this pin inputs the card control register access (/CARDREG). For Toshiba TX3912 Bus, this pin inputs the card control register (CARDREG*). For PowerPC Bus, this pin inputs the system address bit 12 (A12). For all other busses, this pin inputs the system address bit 19 (A19). See "Table 5-7 CPU Interface Pin Mapping" for summary. See the respective AC Timing diagram for detailed functionality.

Table 5-1 Host Interface Pin Descriptions

Pin Name	Type	Pin #	Driver	Reset# State	Description
AB20	I	111	C	Hi-Z	<ul style="list-style-type: none"> For the MIPS/ISA Bus, this pin inputs system address bit 20. Note that for the ISA Bus, the unlatched LA20 must first be latched before input to AB20. For Philips PR31500/31700 Bus, this pin inputs the address latch enable (ALE). For Toshiba TX3912 Bus, this pin inputs the address latch enable (ALE). For PowerPC Bus, this pin inputs the system address bit 11 (A11). For all other busses, this pin inputs the system address bit 20 (A20). See "Table 5-7 CPU Interface Pin Mapping" for summary. See the respective AC Timing diagram for detailed functionality.
DB[15:0]	IO	16–31	C/TS2	Hi-Z	These pins are the system data bus. For 8-bit bus modes, unused data pins should be tied to VDD. <ul style="list-style-type: none"> For SH-3/SH-4 Bus, these pins are connected to D[15:0]. For MC68K Bus 1, these pins are connected to D[15:0]. For MC68K Bus 2, these pins are connected to D[31:16] for 32-bit devices (e.g. MC68030) or D[15:0] for 16-bit devices (e.g. MC68340). For Generic Bus, these pins are connected to D[15:0]. For MIPS/ISA Bus, these pins are connected to SD[15:0]. For Philips PR31500/31700 Bus, these pins are connected to D[31:16]. For Toshiba TX3912 Bus, pins [15:8] are connected to D[23:16] and pins [7:0] are connected to D[31:24]. For PowerPC Bus, these pins are connected to D[0:15]. For PC Card (PCMCIA) Bus, these pins are connected to D[15:0]. See "Table 5-7 CPU Interface Pin Mapping" for summary. See the respective AC Timing diagram for detailed functionality.
WE1#	IO	9	CS/TS2	Hi-Z	This is a multi-purpose pin: <ul style="list-style-type: none"> For SH-3/SH-4 Bus, this pin inputs the write enable signal for the upper data byte (WE1#). For MC68K Bus 1, this pin inputs the upper data strobe (UDS#). For MC68K Bus 2, this pin inputs the data strobe (DS#). For Generic Bus, this pin inputs the write enable signal for the upper data byte (WE1#). For MIPS/ISA Bus, this pin inputs the system byte high enable signal (SBHE#). For Philips PR31500/31700 Bus, this pin inputs the odd byte access enable signal (/CARDxCSH). For Toshiba TX3912 Bus, this pin inputs the odd byte access enable signal (CARDxCSH*). For PowerPC Bus, this pin outputs the burst inhibit signal (BI#). For PC Card (PCMCIA) Bus, this pin inputs the card enable 2 signal (-CE2). See "Table 5-7 CPU Interface Pin Mapping" for summary. See the respective AC Timing diagram for detailed functionality.
M/R#	I	5	C	Hi-Z	<ul style="list-style-type: none"> For Philips PR31500/31700 Bus, this pin is connected to VDD. For Toshiba TX3912 Bus, this pin is connected to VDD. For all busses, this input pin is used to select between the display buffer and register address spaces of the S1D13505. M/R# is set high to access the display buffer and low to access the registers. See Register Mapping. See "Table 5-7 CPU Interface Pin Mapping" on page 1-20.
CS#	I	4	C	Hi-Z	<ul style="list-style-type: none"> For Philips PR31500/31700 Bus, this pin is connected to VDD. For Toshiba TX3912 Bus, this pin is connected to VDD. For all busses, this is the Chip Select input. See "Table 5-7 CPU Interface Pin Mapping" on page 1-20. See the respective AC Timing diagram for detailed functionality.

Table 5-1 Host Interface Pin Descriptions

Pin Name	Type	Pin #	Driver	Reset# State	Description
BUSCLK	I	13	C	Hi-Z	<p>This pin inputs the system bus clock. It is possible to apply a 2x clock and divide it by 2 internally - see MD12 in Summary of Configuration Options.</p> <ul style="list-style-type: none"> For SH-3/SH-4 Bus, this pin is connected to CKIO. For MC68K Bus 1, this pin is connected to CLK. For MC68K Bus 2, this pin is connected to CLK. For Generic Bus, this pin is connected to BCLK. For MIPS/ISA Bus, this pin is connected to CLK. For Philips PR31500/31700 Bus, this pin is connected to DCLKOUT. For Toshiba TX3912 Bus, this pin is connected to DCLKOUT. For PowerPC Bus, this pin is connected to CLKOUT. For PC Card (PCMCIA) Bus, this pin is connected to the input clock (CLKI, pin 69). <p>See "Table 5-7 CPU Interface Pin Mapping" for summary. See the respective AC Timing diagram for detailed functionality.</p>
BS#	I	6	CS	Hi-Z	<p>This is a multi-purpose pin:</p> <ul style="list-style-type: none"> For SH-3/SH-4 Bus, this pin inputs the bus start signal (BS#). For MC68K Bus 1, this pin inputs the address strobe (AS#). For MC68K Bus 2, this pin inputs the address strobe (AS#). For Generic Bus, this pin is connected to VDD. For MIPS/ISA Bus, this pin is connected to VDD. For Philips PR31500/31700 Bus, this pin is connected to VDD. For Toshiba TX3912 Bus, this pin is connected to VDD. For PowerPC Bus, this pin inputs the Transfer Start signal (TS#). For PC Card (PCMCIA) Bus, this pin is connected to VDD. <p>See "Table 5-7 CPU Interface Pin Mapping" for summary. See the respective AC Timing diagram for detailed functionality.</p>
RD/WR#	I	10	CS	Hi-Z	<p>This is a multi-purpose pin:</p> <ul style="list-style-type: none"> For SH-3/SH-4 Bus, this pin inputs the read write signal (RD/WR#). The S1D13505 needs this signal for early decode of the bus cycle. For MC68K Bus 1, this pin inputs the read write signal (R/W#). For MC68K Bus 2, this pin inputs the read write signal (R/W#). For Generic Bus, this pin inputs the read command for the upper data byte (RD1#). For MIPS/ISA Bus, this pin is connected to VDD. For Philips PR31500/31700 Bus, this pin inputs the even byte access enable signal (/CARDxCSL). For Toshiba TX3912 Bus, this pin inputs the even byte access enable signal (CARDxCSL*). For PowerPC Bus, this pin inputs the read write signal (RD/WR#). For PC Card (PCMCIA) Bus, this pin inputs the card enable 1 signal (-CE1). <p>See "Table 5-7 CPU Interface Pin Mapping" for summary. See the respective AC Timing diagram for detailed functionality.</p>
RD#	I	7	CS	Hi-Z	<p>This is a multi-purpose pin:</p> <ul style="list-style-type: none"> For SH-3/SH-4 Bus, this pin inputs the read signal (RD#). For MC68K Bus 1, this pin is connected to VDD. For MC68K Bus 2, this pin inputs the bus size bit 1 (SIZ1). For Generic Bus, this pin inputs the read command for the lower data byte (RD0#). For MIPS/ISA Bus, this pin inputs the memory read signal (MEMR#). For Philips PR31500/31700 Bus, this pin inputs the memory read command (/RD). For Toshiba TX3912 Bus, this pin inputs the memory read command (RD*). For PowerPC Bus, this pin inputs the transfer size 0 signal (TSIZ0). For PC Card (PCMCIA) Bus, this pin inputs the output enable signal (-OE). <p>See "Table 5-7 CPU Interface Pin Mapping" for summary. See the respective AC Timing diagram for detailed functionality.</p>

Table 5-1 Host Interface Pin Descriptions

Pin Name	Type	Pin #	Driver	Reset# State	Description
WE0#	I	8	CS	Hi-Z	<p>This is a multi-purpose pin:</p> <ul style="list-style-type: none"> • For SH-3/SH-4 Bus, this pin inputs the write enable signal for the lower data byte (WE0#). • For MC68K Bus 1, this pin must be connected to VDD • For MC68K Bus 2, this pin inputs the bus size bit 0 (SIZ0). • For Generic Bus, this pin inputs the write enable signal for the lower data byte (WE0#). • For MIPS/ISA Bus, this pin inputs the memory write signal (MEMW#). • For Philips PR31500/31700 Bus, this pin inputs the memory write command (/WE). • For Toshiba TX3912 Bus, this pin inputs the memory write command (WE*). • For PowerPC Bus, this pin inputs the Transfer Size 1 signal (TSIZ1). • For PC Card (PCMCIA) Bus, this pin inputs the write enable signal (-WE). <p>See “Table 5-7 CPU Interface Pin Mapping” for summary. See the respective AC Timing diagram for detailed functionality.</p>
WAIT#	O	15	TS2	Hi-Z	<p>The active polarity of the WAIT# output is configurable; the state of MD5 on the rising edge of RESET# defines the active polarity of WAIT# - see “Summary of Configuration Options”.</p> <ul style="list-style-type: none"> • For SH-3 Bus, this pin outputs the wait request signal (WAIT#); MD5 must be pulled low during reset by the internal pull-down resistor. • For SH-4 Bus, this pin outputs the ready signal (RDY#); MD5 must be pulled high during reset by an external pull-up resistor. • For MC68K Bus 1, this pin outputs the data transfer acknowledge signal (DTACK#); MD5 must be pulled high during reset by an external pull-up resistor. • For MC68K Bus 2, this pin outputs the data transfer and size acknowledge bit 1 (DSACK1#); MD5 must be pulled high during reset by an external pull-up resistor. • For Generic Bus, this pin outputs the wait signal (WAIT#); MD5 must be pulled high during reset by an external pull-up resistor. • For MIPS/ISA Bus, this pin outputs the IO channel ready signal (IOCHRDY); MD5 must be pulled low during reset by the internal pull-down resistor. • For Philips PR31500/31700 Bus, this pin outputs the wait state signal (/CARDxWAIT); MD5 must be pulled low during reset by the internal pull-down resistor. • For Toshiba TX3912 Bus, this pin outputs the wait state signal (CARDx-WAIT*); MD5 must be pulled low during reset by the internal pull-down resistor. • For PowerPC Bus, this pin outputs the transfer acknowledge signal (TA#); MD5 must be pulled high during reset by an external pull-up resistor. • For PC Card (PCMCIA) Bus, this pin outputs the wait signal (-WAIT); MD5 must be pulled low during reset by the internal pull-down resistor. <p>See “Table 5-7 CPU Interface Pin Mapping” for summary. See the respective AC Timing diagram for detailed functionality.</p>
RESET#	I	11	CS	–	<p>Active low input that clears all internal registers and forces all outputs to their inactive states. Note that active high RESET signals must be inverted before input to this pin.</p>

Memory Interface

Table 5-2 Memory Interface Pin Descriptions

Pin Name	Type	Pin #	Driver	Reset# State	Description
LCAS#	O	51	CO1	1	<ul style="list-style-type: none"> For dual-CAS# DRAM, this is the column address strobe for the lower byte (LCAS#). For single-CAS# DRAM, this is the column address strobe (CAS#). See "Table 5-8 Memory Interface Pin Mapping" for summary. See Memory Interface Timing for detailed functionality.
UCAS#	O	52	CO1	1	This is a multi-purpose pin: <ul style="list-style-type: none"> For dual-CAS# DRAM, this is the column address strobe for the upper byte (UCAS#). For single-CAS# DRAM, this is the write enable signal for the upper byte (UWE#). See "Table 5-8 Memory Interface Pin Mapping" for summary. See Memory Interface Timing for detailed functionality.
WE#	O	53	CO1	1	<ul style="list-style-type: none"> For dual-CAS# DRAM, this is the write enable signal (WE#). For single-CAS# DRAM, this is the write enable signal for the lower byte (LWE#). See "Table 5-8 Memory Interface Pin Mapping" for summary. See Memory Interface Timing for detailed functionality.
RAS#	O	54	CO1	1	Row address strobe - see Memory Interface Timing for detailed functionality.
MD[15:0]	IO	34, 36, 38, 40, 42, 44, 46, 48, 49, 47, 45, 43, 41, 39, 37, 35	C/TS1D	Hi-Z	<ul style="list-style-type: none"> Bi-Directional memory data bus. During reset, these pins are inputs and their states at the rising edge of RESET# are used to configure the chip - see Summary of Configuration Options. Internal pull-down resistors (typical values of 100KΩ/180KΩ at 5V/3.3V respectively) pull the reset states to 0. External pull-up resistors can be used to pull the reset states to 1. See Memory Interface Timing for detailed functionality.
MA[8:0]	O	58, 60, 62, 64, 66, 67, 65, 63, 61	CO1	Output	Multiplexed memory address - see Memory Interface Timing for functionality.
MA9	IO	56	C/TS1	Output	This is a multi-purpose pin: <ul style="list-style-type: none"> For 2M byte DRAM, this is memory address bit 9 (MA9). For asymmetrical 512K byte DRAM, this is memory address bit 9 (MA9). For symmetrical 512K byte DRAM, this pin can be used as general purpose IO pin 3 (GPIO3). Note that unless configured otherwise, this pin defaults to an input and must be driven to a valid logic level. See "Table 5-8 Memory Interface Pin Mapping" for summary. See Memory Interface Timing for detailed functionality.
MA10	IO	59	C/TS1	Output	This is a multi-purpose pin: <ul style="list-style-type: none"> For asymmetrical 2M byte DRAM this is memory address bit 10 (MA10). For symmetrical 2M byte DRAM and all 512K byte DRAM this pin can be used as general purpose IO pin 1 (GPIO1). Note that unless configured otherwise, this pin defaults to an input and must be driven to a valid logic level. See "Table 5-8 Memory Interface Pin Mapping" for summary. See Memory Interface Timing for detailed functionality.
MA11	IO	57	C/TS1	Output	This is a multi-purpose pin: <ul style="list-style-type: none"> For asymmetrical 2M byte DRAM this is memory address bit 11 (MA11). For symmetrical 2M byte DRAM and all 512K byte DRAM this pin can be used as general purpose IO pin 2 (GPIO2). Note that unless configured otherwise, this pin defaults to an input and must be driven to a valid logic level. See "Table 5-8 Memory Interface Pin Mapping" for summary. See Memory Interface Timing for detailed functionality.

LCD Interface

Table 5-3 LCD Interface Pin Descriptions

Pin Name	Type	Pin #	Cell	RESET# State	Description
FPDAT[15:0]	O	95–88, 86–79	CN3	Output	Panel data bus. Not all pins are used for some panels - see “Table 5-9 LCD Interface Pin Mapping” for details. Unused pins are driven low.
FPFRAME	O	73	CN3	Output	Frame pulse
FPLINE	O	74	CN3	Output	Line pulse
FPSHIFT	O	77	CO3	Output	Shift clock
LCDPWR	O	75	CO1	Output if MD[10]=0 1 if MD[10]=1	LCD power control output. The active polarity of this output is selected by the state of MD10 at the rising edge of RESET# - see “5.3 Summary of Configuration Options”. This output is controlled by the power save mode circuitry - see “15 Power Save Modes” for details.
DRDY	O	76	CN3	Output	This is a multi-purpose pin: <ul style="list-style-type: none"> • For TFT/D-TFD panels this is the display enable output (DRDY). • For passive LCD with Format 1 interface this is the 2nd Shift Clock (FPSHIFT2). • For all other LCD panels this is the LCD backplane bias signal (MOD). See “Table 5-9 LCD Interface Pin Mapping” and REG[02h] for details.

CRT Interface

Table 5-4 Clock Input Pin Description

Pin Name	Type	Pin #	Cell	RESET# State	Description
HRTC	IO	107	CN3	Output	Horizontal retrace signal for CRT
VRTC	IO	108	CN3	Output	Vertical retrace signal for CRT
RED	O	100	A		Analog output for CRT color Red
GREEN	O	103	A		Analog output for CRT color Green
BLUE	O	105	A		Analog output for CRT color Blue
IREF	I	101	A		Current reference for DAC - see Analog Pins. This pin must be left unconnected if the DAC is not needed.

Miscellaneous

Table 5-5 Miscellaneous Interface Pin Descriptions

Pin Name	Type	Pin #	Cell	RESET# State	Description
SUSPEND#	IO	71	CS/TS1	Hi-Z if MD[9]=0 High if MD[10:9]=01 Low if MD[10:9]=11	This pin can be used as a power-down input (SUSPEND#) or as an output possibly used for controlling the LCD backlight power: <ul style="list-style-type: none"> When MD9 = 0 at rising edge of RESET#, this pin is an active-low Schmitt input used to put the S1D13505 into Hardware suspend mode - see "15 Power Save Modes" for details. When MD[10:9] = 01 at rising edge of RESET#, this pin is an output (GPO) with a reset state of 1. Its state is controlled by REG[21h] bit 7. When MD[10:9] = 11 at rising edge of RESET#, this pin is an output (GPO) with a reset state of 0. Its state is controlled by REG[21h] bit 7.
CLKI	I	69	C		Input clock for the internal pixel clock (PCLK) and memory clock (MCLK). PCLK and MCLK are derived from CLKI - see REG[19h] for details.
TESTEN	I	70	CD	Hi-Z	Test Enable. This pin should be connected to Vss for normal operation.
VDD	P	12, 33, 55, 72, 97, 109	P		VDD
DACVDD	P	99, 102, 104	P		DAC VDD
Vss	P	14, 32, 50, 68, 78, 87, 96, 110	P		Vss
DACVss	P	98, 106	P		DAC Vss

5.3 Summary of Configuration Options

Table 5-6 Summary of Power On / Reset Options

Pin Name	Value on this pin at rising edge of RESET# is used to configure: (1/0)	
	1	0
MD0	8-bit host bus interface	16-bit host bus interface
MD[3:1]	Select host bus interface: MD[11] = 0: 000 = SH-3/SH-4 bus interface 001 = MC68K Bus 1 010 = MC68K Bus 2 011 = Generic 100 = Reserved 101 = MIPS/ISA 110 = PowerPC 111 = PC Card (when MD11 = 1 Philips PR31500/PR31700 or Toshiba TX3912 Bus)	
MD4	Little Endian	Big Endian
MD5	WAIT# is active high (1 = insert wait state)	WAIT# is active low (0 = insert wait state)
MD[7:6]	Memory Address/GPIO configuration: 00 = symmetrical 256K×16 DRAM. MA[8:0] = DRAM address. MA[11:9] = GPIO2,1,3 pins. 01 = symmetrical 1M×16 DRAM. MA[9:0] = DRAM address. MA[10:11] = GPIO2,1 pins. 10 = asymmetrical 256K×16 DRAM. MA[9:0] = DRAM address. MA[10:11] = GPIO2,1 pins. 11 = asymmetrical 1M×16 DRAM. MA[11:0] = DRAM address.	
MD8	Not used	
MD9	SUSPEND# pin configured as GPO output	SUSPEND# pin configured as SUSPEND# input
MD10	Active low LCDPWR and GPO polarities	Active high LCDPWR and GPO polarities
MD11	Alternate Host Bus Interface Selected	Primary Host Bus Interface Selected
MD12	BUSCLK input divided by 2	BUSCLK input not divided
MD[15:13]	Not used	

5.4 Multiple Function Pin Mapping

Table 5-7 CPU Interface Pin Mapping

S1D13505 Pin Name	SH-3	SH-4	MC68K Bus 1	MC68K Bus 2	Generic	MIPS/ISA	Philips PR31500 /PR31700	Toshiba TX3912	PowerPC	PC Card (PCMCIA)
AB20	A20	A20	A20	A20	A20	LatchA20	ALE	ALE	A11	A20
AB19	A19	A19	A19	A19	A19	SA19	/CAR- DREG	CAR- DREG*	A12	A19
AB18	A18	A18	A18	A18	A18	SA18	/CAR- DIORD	CAR- DIORD*	A13	A18
AB17	A17	A17	A17	A17	A17	SA17	/CAR- DIOWR	CAR- DIOWR*	A14	A17
AB[16:13]	A[16:13]	A[16:13]	A[16:13]	A[16:13]	A[16:13]	SA[16:13]	VDD	VDD	A[15:18]	A[16:13]
AB[12:1]	A[12:1]	A[12:1]	A[12:1]	A[12:1]	A[12:1]	SA[12:1]	A[12:1]	A[12:1]	A[19:30]	A[12:1]
AB0	A0	A0	LDS#	A0	A0	SA0	A0	A0	A31	A0
DB[15:8]	D[15:8]	D[15:8]	D[15:8]	D[31:24]	D[15:8]	SD[15:8]	D[31:24]	D[31:24]	D[0:7]	D[15:8]
DB[7:0]	D[7:0]	D[7:0]	D[7:0]	D[23:16]	D[7:0]	SD[7:0]	D[23:16]	D[23:16]	D[8:15]	D[7:0]
WE1#	WE1#	WE1#	UDS#	DS#	WE1#	SBHE#	/CARD xCSH	CARD xCSH*	BI#	-CE2
M/R#	External Decode						VDD		External Decode	
CS#	External Decode						VDD		External Decode	
BUSCLK	CKIO	CKIO	CLK	CLK	BCLK	CLK	DCLK- OUT	DCLK- OUT	CLKOUT	CLKI
BS#	BS#	BS#	AS#	AS#	VDD	VDD	VDD	VDD	TS#	VDD
RD/WR#	RD/WR#	RD/WR#	R/W#	R/W#	RD1#	VDD	/CARD xCSL	CARD xCSL*	RD/WR#	-CE1
RD#	RD#	RD#	VDD	SIZ1	RD0#	MEMR#	/RD	RD*	TSIZ0	-OE
WE0#	WE0#	WE0#	VDD	SIZ0	WE0#	MEMW#	/WE	WE*	TSIZ1	-WE
WAIT#	WAIT#	RDY	DTACK#	DSACK1#	WAIT#	IOCHRD Y	/CARD x WAIT	CARD x WAIT*	TA#	-WAIT
RESET#	RESET#	RESET#	RESET#	RESET#	RESET#	inverted RESET	RESET#	PON*	RESET#	inverted RESET

Table 5-8 Memory Interface Pin Mapping

S1D13505 Pin Name	FPM/EDO-DRAM							
	Sym 256Kx16		Asym 256Kx16		Sym 1Mx16		Asym 1Mx16	
	2-CAS#	2-WE#	2-CAS#	2-WE#	2-CAS#	2-WE#	2-CAS#	2-WE#
MD[15:0]	D[15:0]							
MA[8:0]	A[8:0]							
MA9	GPIO3		A9				A9	
MA10	GPIO1						A10	
MA11	GPIO2						A11	
UCAS#	UCAS#	UWE#	UCAS#	UWE#	UCAS#	UWE#	UCAS#	UWE#
LCAS#	LCAS#	CAS#	LCAS#	CAS#	LCAS#	CAS#	LCAS#	CAS#
WE#	WE#	LWE#	WE#	LWE#	WE#	LWE#	WE#	LWE#
RAS#	RAS#							

Notes: • All GPIO pins default to input on reset and unless programmed otherwise, should be connected to either Vss or IO VDD if not used.

- The bus signal A0 is not used by the S1D13505 internally.

Table 5-9 LCD Interface Pin Mapping

S1D13505 Pin Name	Monochrome Passive Panel			Color Passive Panel						Color TFT/D-TFD Panel		
	Single		Dual	Single	Single Format 1	Single Format 2	Single	Dual				
	4-bit	8-bit	8-bit	4-bit	8-bit	8-bit	16-bit	8-bit	16-bit	9-bit	12-bit	18-bit
FPFRAME	FPFRAME											
FPLINE	FPLINE											
FPSHIFT	FPSHIFT											
DRDY	MOD				FPSHIFT 2	MOD				DRDY		
FPDAT0	driven 0	D0	LD0	driven 0	D0	D0	D0	LD0	LD0	R2	R3	R5
FPDAT1	driven 0	D1	LD1	driven 0	D1	D1	D1	LD1	LD1	R1	R2	R4
FPDAT2	driven 0	D2	LD2	driven 0	D2	D2	D2	LD2	LD2	R0	R1	R3
FPDAT3	driven 0	D3	LD3	driven 0	D3	D3	D3	LD3	LD3	G2	G3	G5
FPDAT4	D0	D4	UD0	D0	D4	D4	D4	UD0	UD0	G1	G2	G4
FPDAT5	D1	D5	UD1	D1	D5	D5	D5	UD1	UD1	G0	G1	G3
FPDAT6	D2	D6	UD2	D2	D6	D6	D6	UD2	UD2	B2	B3	B5
FPDAT7	D3	D7	UD3	D3	D7	D7	D7	UD3	UD3	B1	B2	B4
FPDAT8	driven 0	driven 0	driven 0	driven 0	driven 0	driven 0	D8	driven 0	LD4	B0	B1	B3
FPDAT9	driven 0	driven 0	driven 0	driven 0	driven 0	driven 0	D9	driven 0	LD5	driven 0	R0	R2
FPDAT10	driven 0	driven 0	driven 0	driven 0	driven 0	driven 0	D10	driven 0	LD6	driven 0	driven 0	R1
FPDAT11	driven 0	driven 0	driven 0	driven 0	driven 0	driven 0	D11	driven 0	LD7	driven 0	G0	G2
FPDAT12	driven 0	driven 0	driven 0	driven 0	driven 0	driven 0	D12	driven 0	UD4	driven 0	driven 0	G1
FPDAT13	driven 0	driven 0	driven 0	driven 0	driven 0	driven 0	D13	driven 0	UD5	driven 0	driven 0	G0
FPDAT14	driven 0	driven 0	driven 0	driven 0	driven 0	driven 0	D14	driven 0	UD6	driven 0	B0	B2
FPDAT15	driven 0	driven 0	driven 0	driven 0	driven 0	driven 0	D15	driven 0	UD7	driven 0	driven 0	B1

5.5 CRT Interface

The following figure shows the external circuitry for the CRT interface.

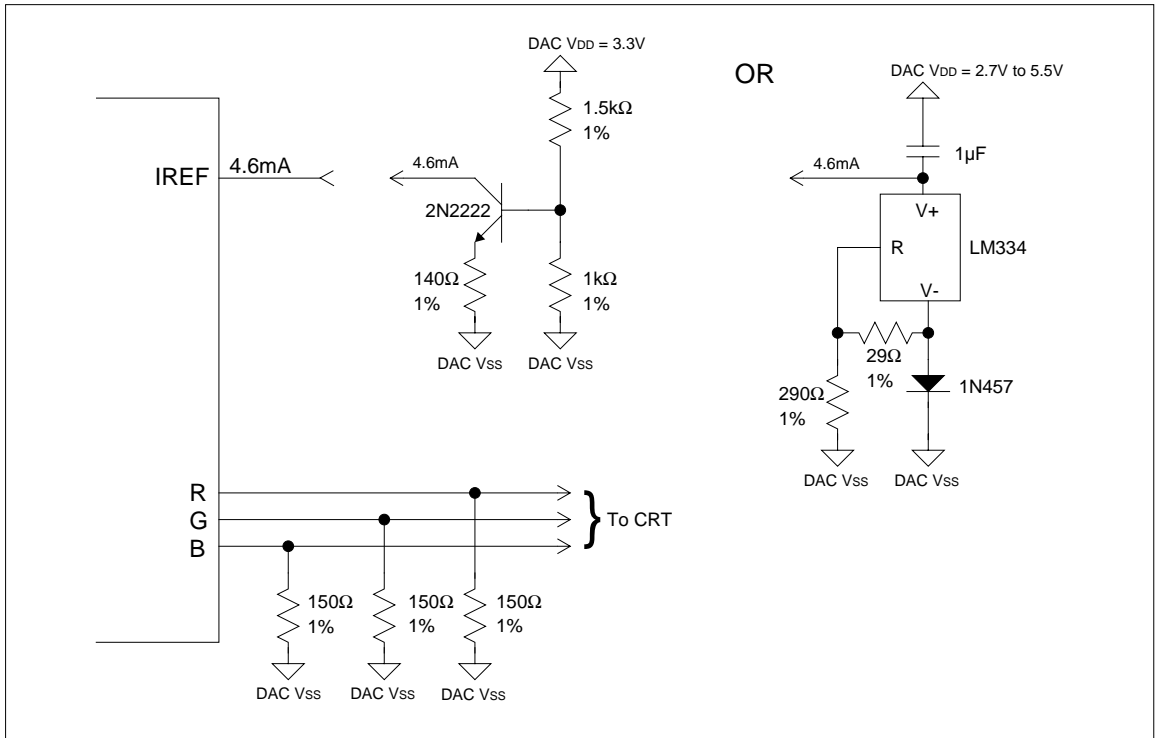


Figure 5-2 External Circuitry for CRT Interface

6 D.C. CHARACTERISTICS

Table 6-1 Absolute Maximum Ratings

Symbol	Parameter	Rating	Units
V _{DD}	Supply Voltage	V _{SS} - 0.3 to 6.0	V
DAC V _{DD}	Supply Voltage	V _{SS} - 0.3 to 6.0	V
V _{IN}	Input Voltage	V _{SS} - 0.3 to V _{DD} + 0.5	V
V _{OUT}	Output Voltage	V _{SS} - 0.3 to V _{DD} + 0.5	V
T _{STG}	Storage Temperature	-65 to 150	°C
T _{SOL}	Solder Temperature/Time	260 for 10 sec. max at lead	°C

Table 6-2 Recommended Operating Conditions

Symbol	Parameter	Condition	Min.	Typ.	Max.	Units
V _{DD}	Supply Voltage	V _{SS} = 0V	2.7	3.0/3.3/5.0	5.5	V
V _{IN}	Input Voltage		V _{SS}		V _{DD}	V
T _{OPR}	Operating Temperature		-40	25	85	°C

Table 6-3 Electrical Characteristics for V_{DD} = 5.0V Typical

Symbol	Parameter	Condition	Min.	Typ.	Max.	Units
I _{DD}	Quiescent Current	Quiescent Conditions			400	μA
I _{IZ}	Input Leakage Current		-1		1	μA
I _{OZ}	Output Leakage Current		-1		1	μA
V _{OH}	High Level Output Voltage	V _{DD} = min I _{OL} = -4mA (Type1), -8mA (Type2), -12mA (Type3)	V _{DD} - 0.4			V
V _{OL}	Low Level Output Voltage	V _{DD} = min I _{OL} = 4mA (Type1), 8mA (Type2), 12mA (Type3)			0.4	V
V _{IH}	High Level Input Voltage	CMOS level, V _{DD} = max	3.5			V
V _{IL}	Low Level Input Voltage	CMOS level, V _{DD} = min			1.0	V
V _{T+}	High Level Input Voltage	CMOS Schmitt, V _{DD} = 5.0V			4.0	V
V _{T-}	Low Level Input Voltage	CMOS Schmitt, V _{DD} = 5.0V	0.8			V
V _{H1}	Hysteresis Voltage	CMOS Schmitt, V _{DD} = 5.0V	0.3			V
R _{PD}	Pull Down Resistance	V _I = V _{DD}	50	100	200	kΩ
C _I	Input Pin Capacitance				12	pF
C _O	Output Pin Capacitance				12	pF
C _{IO}	Bi-Directional Pin Capacitance				12	pF

Table 6-4 Electrical Characteristics for V_{DD} = 3.3V Typical

Symbol	Parameter	Condition	Min.	Typ.	Max.	Units
I _{DD}	Quiescent Current	Quiescent Conditions			290	μA
I _{IZ}	Input Leakage Current		-1		1	μA
I _{OZ}	Output Leakage Current		-1		1	μA
V _{OH}	High Level Output Voltage	V _{DD} = min I _{OL} = -2mA (Type1), -4mA (Type2), -6mA (Type3)	V _{DD} - 0.3			V
V _{OL}	Low Level Output Voltage	V _{DD} = min I _{OL} = 2mA (Type1), 4mA (Type2), 6mA (Type3)			0.3	V
V _{IH}	High Level Input Voltage	CMOS level, V _{DD} = max	2.2			V
V _{IL}	Low Level Input Voltage	CMOS level, V _{DD} = min			0.8	V
V _{T+}	High Level Input Voltage	CMOS Schmitt, V _{DD} = 3.3V			2.4	V
V _{T-}	Low Level Input Voltage	CMOS Schmitt, V _{DD} = 3.3V	0.6			V
V _{H1}	Hysteresis Voltage	CMOS Schmitt, V _{DD} = 3.3V	0.1			V
R _{PD}	Pull Down Resistance	V _I = V _{DD}	90	180	360	kΩ
C _I	Input Pin Capacitance				12	pF
C _O	Output Pin Capacitance				12	pF
C _{IO}	Bi-Directional Pin Capacitance				12	pF

Table 6-5 Electrical Characteristics for $V_{DD} = 3.0V$ Typical

Symbol	Parameter	Condition	Min.	Typ.	Max.	Units
I_{DDs}	Quiescent Current	Quiescent Conditions			260	μA
I_{IZ}	Input Leakage Current		-1		1	μA
I_{OZ}	Output Leakage Current		-1		1	μA
V_{OH}	High Level Output Voltage	$V_{DD} = \min$ $I_{OL} = -1.8mA$ (Type1), -3.5mA (Type2), -5mA (Type3)	$V_{DD} - 0.3$			V
V_{OL}	Low Level Output Voltage	$V_{DD} = \min$ $I_{OL} = 1.8mA$ (Type1), 3.5mA (Type2), 5mA (Type3)			0.3	V
V_{IH}	High Level Input Voltage	CMOS level, $V_{DD} = \max$	2.0			V
V_{IL}	Low Level Input Voltage	CMOS level, $V_{DD} = \min$			0.8	V
V_{T+}	High Level Input Voltage	CMOS Schmitt, $V_{DD} = 3.0V$			2.3	V
V_{T-}	Low Level Input Voltage	CMOS Schmitt, $V_{DD} = 3.0V$	0.5			V
V_{HI}	Hysteresis Voltage	CMOS Schmitt, $V_{DD} = 3.0V$	0.1			V
R_{PD}	Pull Down Resistance	$V_I = V_{DD}$	100	200	400	$k\Omega$
C_I	Input Pin Capacitance				12	pF
C_O	Output Pin Capacitance				12	pF
C_{IO}	Bi-Directional Pin Capacitance				12	pF

7 A.C. CHARACTERISTICS

Conditions: $V_{DD} = 3.0V \pm 10\%$ and $V_{DD} = 5.0V \pm 10\%$

$T_A = -40^\circ\text{C}$ to 85°C

T_{rise} and T_{fall} for all inputs must be ≤ 5 nsec (10% to 90%)

$C_L = 50\text{pF}$ (CPU Interface), unless noted

$C_L = 100\text{pF}$ (LCD Panel Interface)

$C_L = 10\text{pF}$ (Display Buffer Interface)

$C_L = 10\text{pF}$ (CRT Interface)

7.1 CPU Interface Timing

SH-4 Interface Timing

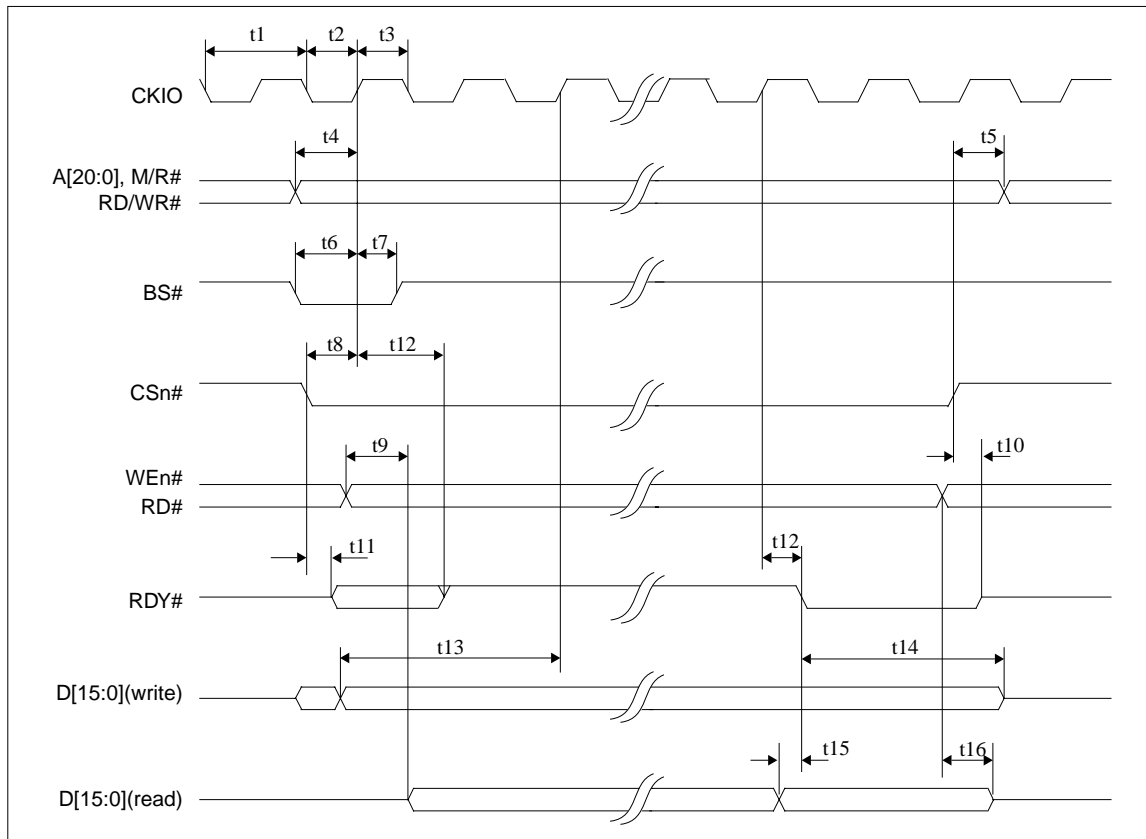


Figure 7-1 SH-4 Timing

- Notes:**
- The above timing diagram is not applicable if the BUSCLK divided by 2 configuration option is selected.
 - The SH-4 Wait State Control Register for the area in which the S1D13505 resides must be set to a non-zero value. The SH-4 read-to-write cycle transition must be set to a non-zero value (with reference to BUSCLK).

Table 7-1 SH-4 Timing

Symbol	Parameter	3.0V ^a		5.0V ^b		Units
		Min.	Max.	Min.	Max.	
t1	Clock period	15		15		ns
t2	Clock pulse width high	6		6		ns
t3	Clock pulse width low	6		6		ns
t4	A[20:0], M/R#, RD/WR# setup to CKIO	3		3		ns
t5	A[20:0], M/R#, RD/WR# hold from CS#	0		0		ns
t6	BS# setup	4		4		ns
t7	BS# hold	1		1		ns
t8	CSn# setup	4		4		ns
t9 ^{#2}	Falling edge RD# to DB[15:0] driven	0		0		ns
t10	Rising edge CSn# to RDY# tri-state	5	25	2.5	10	ns
t11 ^{#1}	Falling edge CSn# to RDY# driven	0	15	0	10	ns
t12	CKIO to WAIT# delay	4	20	3.6	12	ns
t13	DB[15:0] setup to 2 nd CKIO after BS# (write cycle)	10		10		ns
t14	DB[15:0] hold (write cycle)	0		0		ns
t15	DB[15:0] valid to RDY# falling edge (read cycle)	0		0		ns
t16	Rising edge RD# to DB[15:0] tri-state (read cycle)	5	25	2.5	10	ns

a Two Software WAIT States Required

b One Software WAIT State Required

- #1. If the S1D13505 host interface is disabled, the timing for RDY# driven is relative to the falling edge of CSn# or the first positive edge of CKIO after A[20:0], M/R# becomes valid, whichever one is later.
- #2. If the S1D13505 host interface is disabled, the timing for D[15:0] driven is relative to the falling edge of RD# or the first positive edge of CKIO after A[20:0], M/R# becomes valid, whichever one is later.

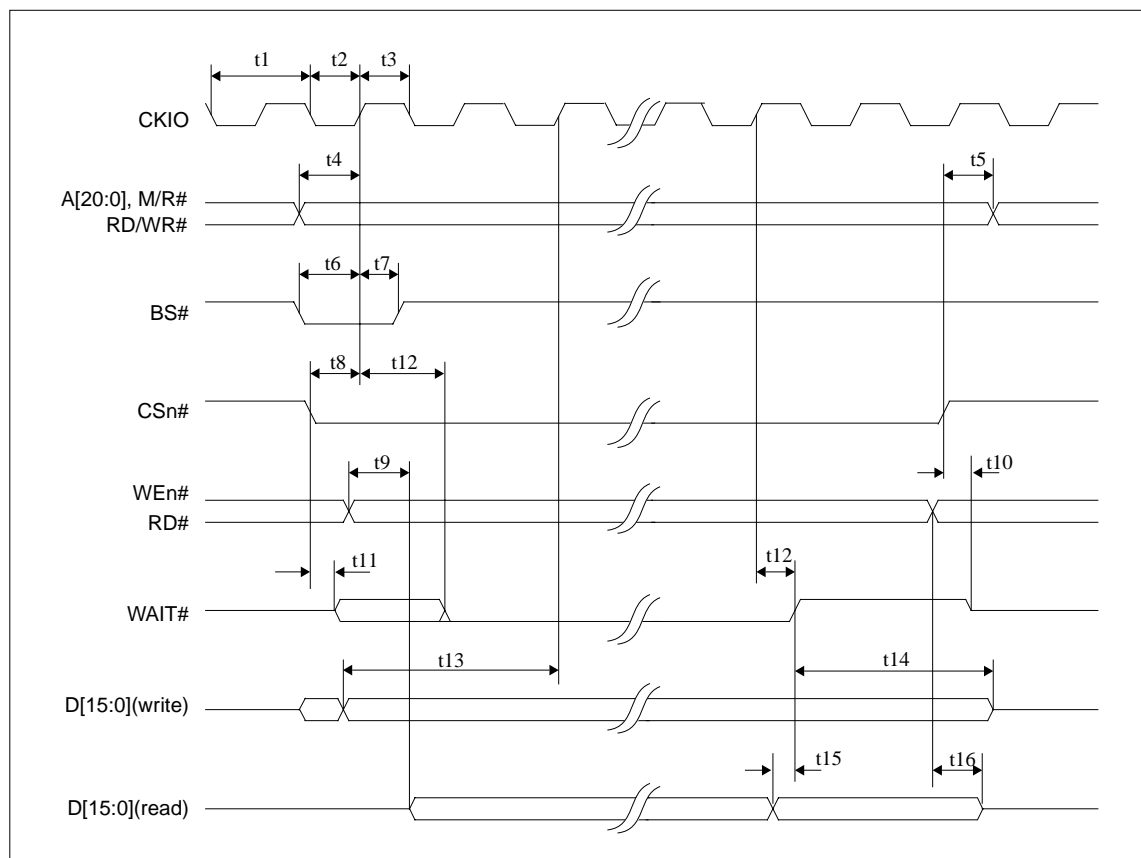
SH-3 Interface Timing

Figure 7-2 SH-3 Timing

- Notes:**
- The above timing diagram is not applicable if the BUSCLK divided by 2 configuration option is selected.
 - The SH-3 Wait State Control Register for the area in which the S1D13505 resides must be set to a non-zero value.

Table 7-2 SH-3 Timing

Symbol	Parameter	3.0V ^a		5.0V ^b		Units
		Min.	Max.	Min.	Max.	
t1	Clock period	16.6		16.6		ns
t2	Clock pulse width high	6		6		ns
t3	Clock pulse width low	6		6		ns
t4	A[20:0], M/R#, RD/WR# setup to CKIO	3		3		ns
t5	A[20:0], M/R#, RD/WR# hold from CS#	0		0		ns
t6	BS# setup	4		4		ns
t7	BS# hold	1		1		ns
t8	CSn# setup	4		4		ns
t9#2	Falling edge RD# to DB[15:0] driven	0		0		ns
t10	Rising edge CSn# to WAIT# tri-state	5	25	2.5	10	ns
t11#1	Falling edge CSn# to WAIT# driven	0	15	0	10	ns
t12	CKIO to WAIT# delay	4	20	3.6	12	ns
t13	DB[15:0] setup to 2 nd CKIO after BS# (write cycle)	10		10		ns
t14	DB[15:0] hold (write cycle)	0		0		ns
t15	DB[15:0] valid to WAIT# rising edge (read cycle)	0		0		ns
t16	Rising edge RD# to DB[15:0] tri-state (read cycle)	5	25	2.5	10	ns

a Two Software WAIT States Required

b One Software WAIT State Required

#1. If the S1D13505 host interface is disabled, the timing for WAIT# driven is relative to the falling edge of CSn# or the first positive edge of CKIO after A[20:0], M/R# becomes valid, whichever one is later.

#2. If the S1D13505 host interface is disabled, the timing for D[15:0] driven is relative to the falling edge of RD# or the first positive edge of CKIO after A[20:0], M/R# becomes valid, whichever one is later.

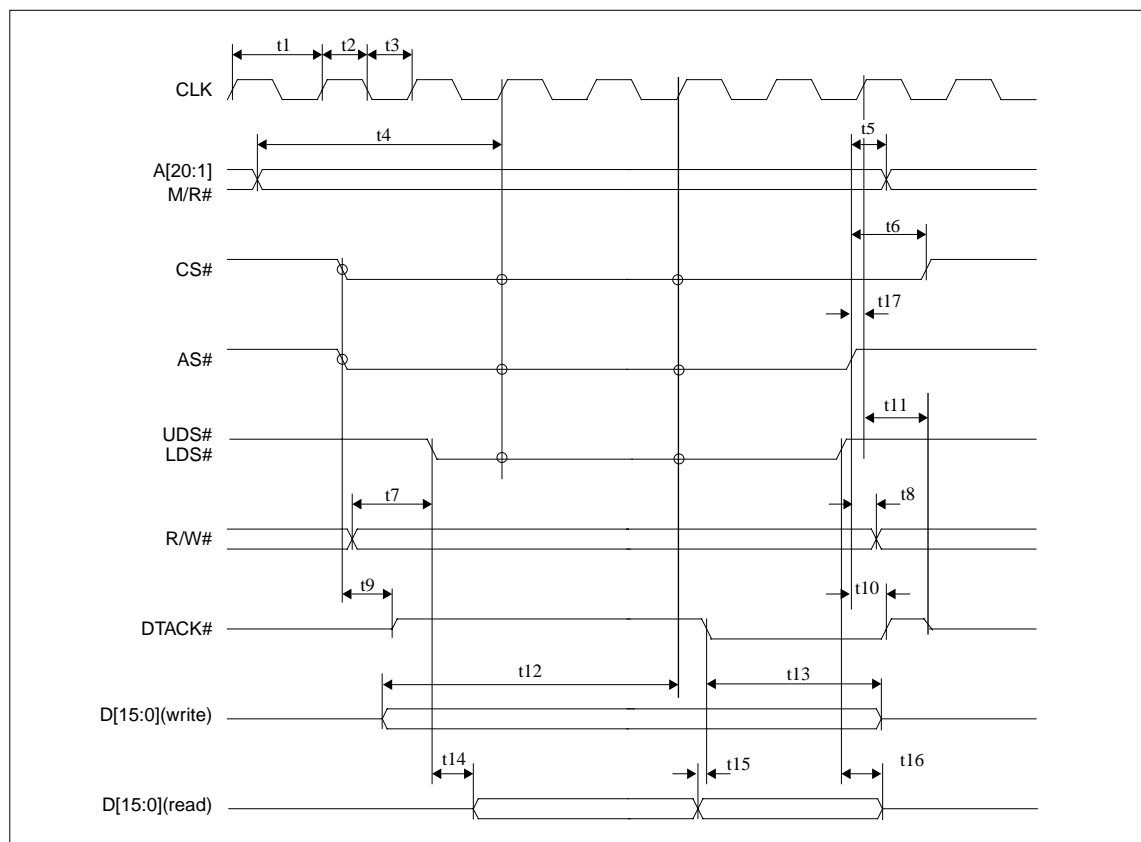
MC68K Bus 1 Interface Timing (e.g. MC68000)

Figure 7-3 MC68000 Timing

Note: The above timing diagram is not applicable if the BUSCLK divided by 2 configuration option is selected.

Table 7-3 MC68000 Timing

Symbol	Parameter	3.3V		5.0V		Units
		Min.	Max.	Min.	Max.	
t1	Clock period	20		20		ns
t2	Clock pulse width high	6		6		ns
t3	Clock pulse width low	6		6		ns
t4	A[20:1], M/R# setup to first CLK where CS# = 0 AS# = 0, and either UDS# = 0 or LDS# = 0	10		10		ns
t5	A[20:1], M/R# hold from AS#	0		0		ns
t6	CS# hold from AS#	0		0		ns
t7	R/W# setup to before to either UDS# = 0 or LDS# = 0	10		10		ns
t8	R/W# hold from AS#	0		0		ns
t9#1	AS# = 0 and CS# = 0 to DTACK# driven high	0		0		ns
t10	AS# high to DTACK# high	3	18	3	12	ns
t11	First BCLK where AS# = 1 to DTACK# high impedance		25		10	ns
t12	D[15:0] valid to third CLK where CS# = 0 AS# = 0, and either UDS# = 0 or LDS# = 0 (write cycle)	10		10		ns
t13	D[15:0] hold from falling edge of DTACK# (write cycle)	0		0		ns
t14#2	Falling edge of UDS# = 0 or LDS# = 0 to DB driven (read cycle)	0		0		ns
t15	D[15:0] valid to DTACK# falling edge (read cycle)	0		0		ns
t16	UDS# and LDS# high to D[15:0] invalid/high impedance (read cycle)	5	25	2.5	10	ns
t17	AS# high setup to CLK	2		2		ns

- #1. If the S1D13505 host interface is disabled, the timing for DTACK# driven high is relative to the falling edge of CS#, AS# or the first positive edge of CLK after A[20:1], M/R# becomes valid, whichever one is later.
- #2. If the S1D13505 host interface is disabled, the timing for D[15:0] driven is relative to the falling edge of UDS#, LDS# or the first positive edge of CLK after A[20:1], M/R# becomes valid, whichever one is later.

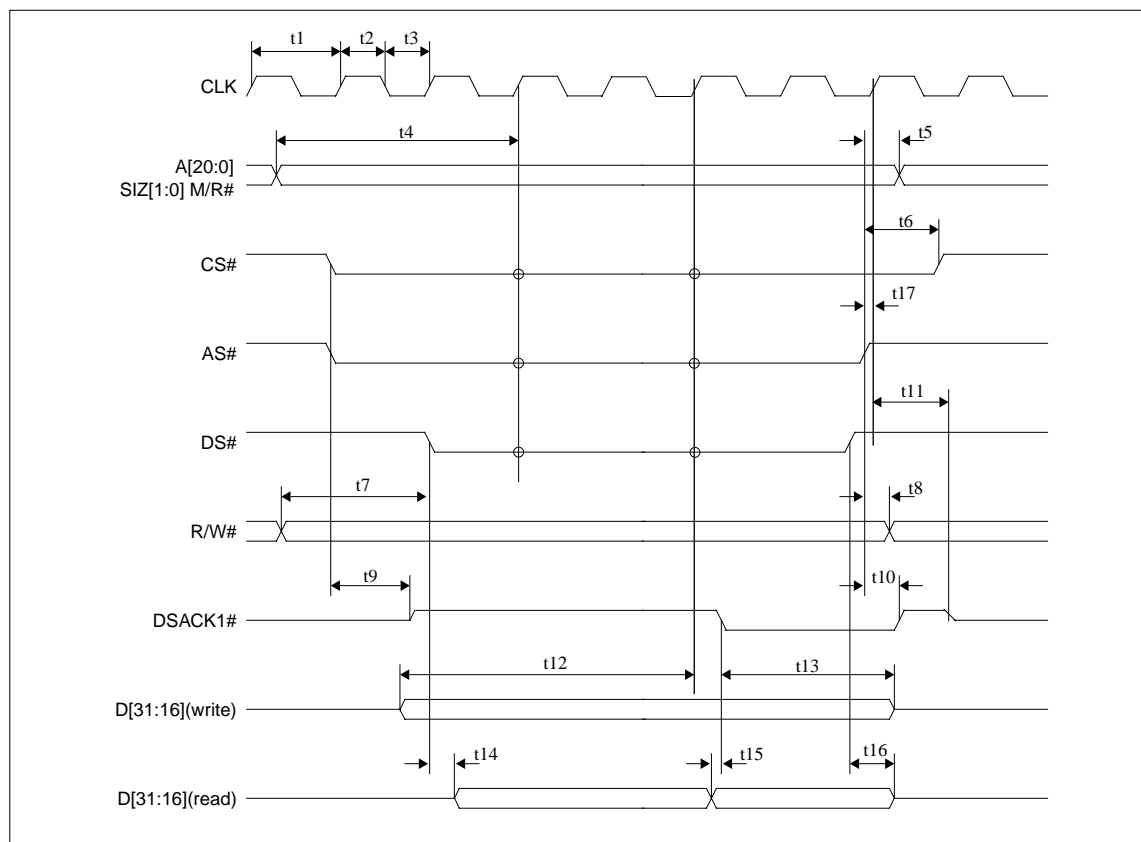
MC68K Bus 2 Interface Timing (e.g. MC68030)

Figure 7-4 MC68030 Timing

Note: The above timing diagram is not applicable if the BUSCLK divided by 2 configuration option is selected.

Table 7-4 MC68030 Timing

Symbol	Parameter	3.3V		5.0V		Units
		Min.	Max.	Min.	Max.	
t1	Clock period	20		20		ns
t2	Clock pulse width high	6		6		ns
t3	Clock pulse width low	6		6		ns
t4	A[20:1], M/R# setup to first CLK where CS# = 0 AS# = 0, and either UDS# = 0 or LDS# = 0	10		10		ns
t5	A[20:1], M/R# hold from AS#	0		0		ns
t6	CS# hold from AS#	0		0		ns
t7	R/W# setup to before to either UDS# = 0 or LDS# = 0	10		10		ns
t8	R/W# hold from AS#	0		0		ns
t9#1	AS# = 0 and CS# = 0 to DTACK# driven high	0		0		ns
t10	AS# high to DTACK# high	3	18	3	12	ns
t11	First BCLK where AS# = 1 to DTACK# high impedance		25		10	ns
t12	D[15:0] valid to third CLK where CS# = 0 AS# = 0, and either UDS# = 0 or LDS# = 0 (write cycle)	10		10		ns
t13	D[15:0] hold from falling edge of DTACK# (write cycle)	0		0		ns
t14#2	Falling edge of UDS# = 0 or LDS# = 0 to DB driven (read cycle)	0		0		ns
t15	D[15:0] valid to DTACK# falling edge (read cycle)	0		0		ns
t16	UDS# and LDS# high to D[15:0] invalid/high impedance (read cycle)	5	25	2.5	10	ns
t17	AS# high setup to CLK	2		2		ns

- #1. If the S1D13505 host interface is disabled, the timing for DSACK1# driven high is relative to the falling edge of CS#, AS# or the first positive edge of CLK after A[20:0], M/R# becomes valid, whichever one is later.
- #2. If the S1D13505 host interface is disabled, the timing for D[31:16] driven is relative to the falling edge of UDS#, LDS# or the first positive edge of CLK after A[20:0], M/R# becomes valid, which ever one is later.

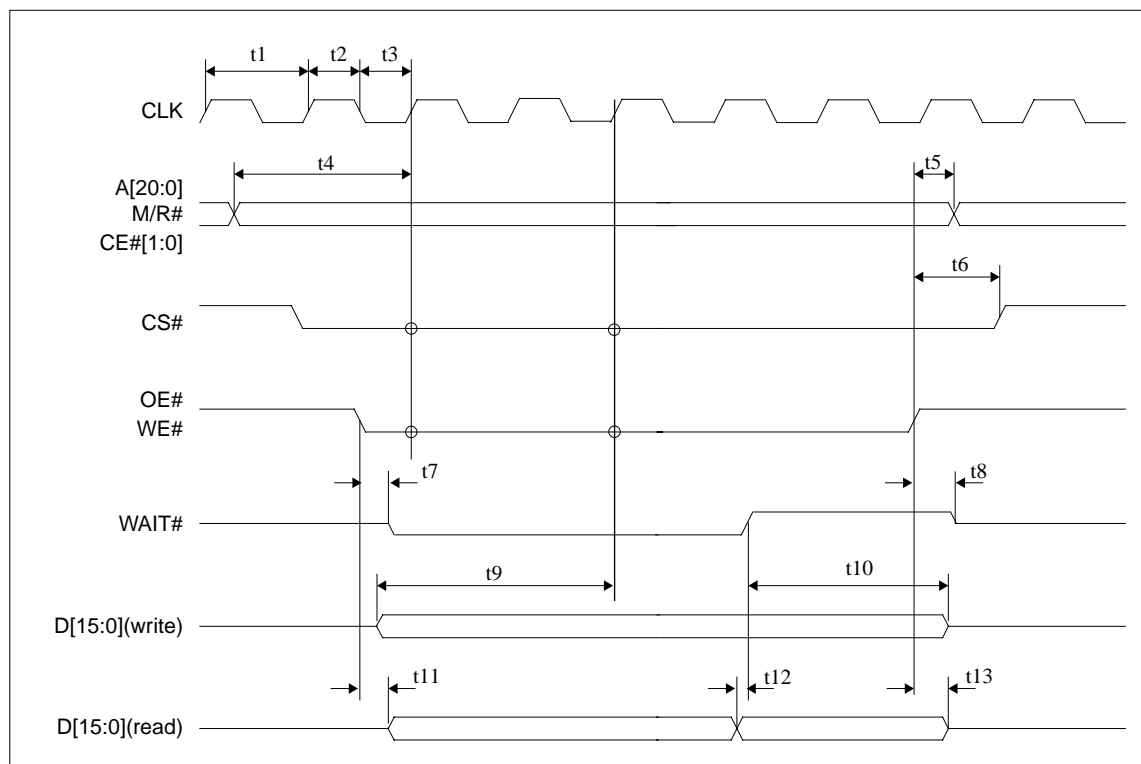
PC Card Interface Timing

Figure 7-5 PC Card Interface Timing

Note: The above timing diagram is not applicable if the BUSCLK divided by 2 configuration option is selected.

Table 7-5 PC Card Interface Timing

Symbol	Parameter	3.0V		5.0V		Units
		Min.	Max.	Min.	Max.	
t1	Clock period	20		20		ns
t2	Clock pulse width high	6		6		ns
t3	Clock pulse width low	6		6		ns
t4	A[20:0], M/R# setup to first CLK where CE# = 0 and either OE# = 0 or WE# = 0	10		10		ns
t5	A[20:0], M/R# hold from rising edge of either OE# or WE#	0		0		ns
t6	CE# hold from rising edge of either OE# or WE#	0		0		ns
t7#1	Falling edge of either OE# or WE# to -WAIT driven low	0	15	0	10	ns
t8	Rising edge of either OE# or WE# to -WAIT tri-state	5	25	2.5	10	ns
t9	D[15:0] setup to third CLK where CE# = 0 and WE# = 0 (write cycle)	10		10		ns
t10	D[15:0] hold (write cycle)	0		0		ns
t11#2	Falling edge OE# to D[15:0] driven (read cycle)	0		0		ns
t12	D[15:0] setup to rising edge WAIT# (read cycle)	0		0		ns
t13	Rising edge of OE# to D[15:0] tri-state (read cycle)	5	25	5	10	ns

- #1. If the S1D13505 host interface is disabled, the timing for WAIT# driven low is relative to the falling edge of OE#, WE# or the first positive edge of CLK after A[20:0], M/R# becomes valid, whichever one is later.
- #2. If the S1D13505 host interface is disabled, the timing for D[15:0] driven is relative to the falling edge of OE# or the first positive edge of CLK after A[20:0], M/R# becomes valid, whichever one is later.

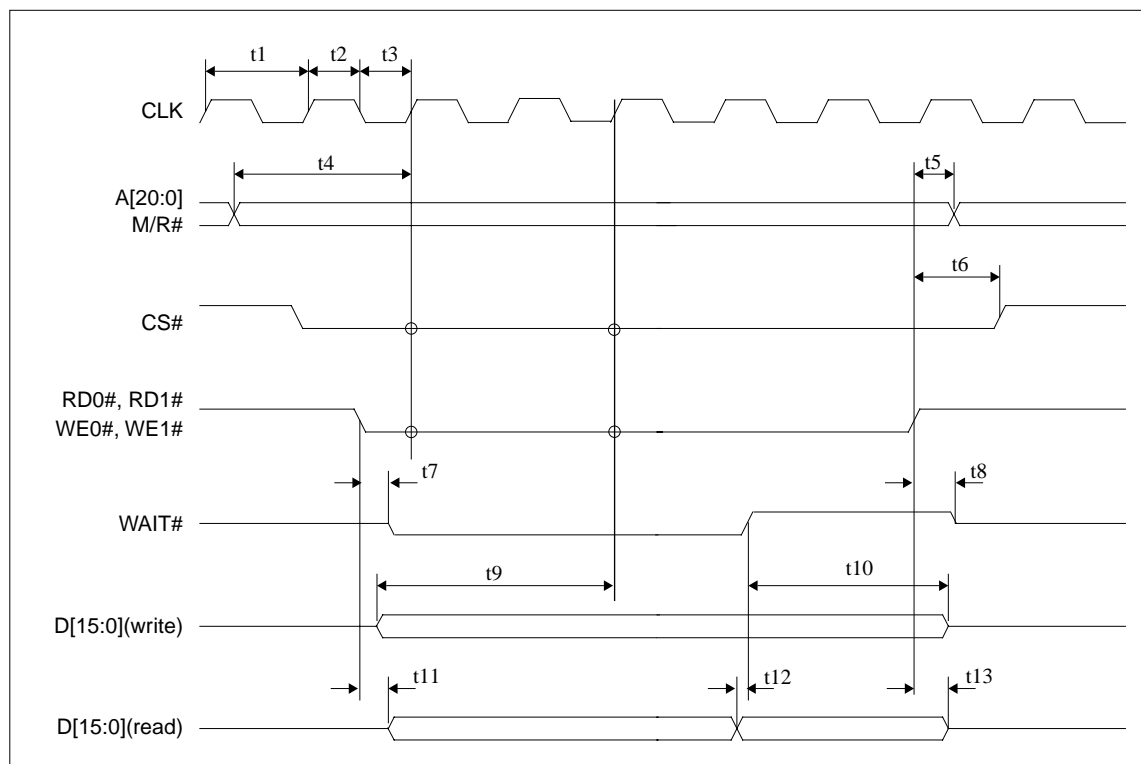
Generic Interface Timing

Figure 7-6 Generic Timing

Note: The above timing diagram is not applicable if the BUSCLK divided by 2 configuration option is selected.

Table 7-6 Generic Timing

Symbol	Parameter	3.0V		5.0V		Units
		Min.	Max.	Min.	Max.	
t1	Clock period	20		20		ns
t2	Clock pulse width high	6		6		ns
t3	Clock pulse width low	6		6		ns
t4	A[20:0], M/R# setup to first CLK where CS# = 0 and either RD0#, RD1#, WE0# or WE1# = 0	10		10		ns
t5	A[20:0], M/R# hold from rising edge of either RD0#, RD1#, WE0# or WE1#	0		0		ns
t6	CS# hold from rising edge of either RD0#, RD1#, WE0# or WE1#	0		0		ns
t7 ^{#1}	Falling edge of either RD0#, RD1#, WE0# or WE1# to WAIT# driven low	0	15	0	10	ns
t8	Rising edge of either RD0#, RD1#, WE0# or WE1# to WAIT# tri-state	5	25	2.5	10	ns
t9	D[15:0] setup to third CLK where CS# = 0 and WE0#, WE1# = 0 (write cycle)	10		10		ns
t10	D[15:0] hold (write cycle)	0		0		ns
t11 ^{#2}	Falling edge RD0#, RD1# to D[15:0] driven (read cycle)	0		0		ns
t12	D[15:0] setup to rising edge WAIT# (read cycle)	0		0		ns
t13	Rising edge of RD0#, RD1# to D[15:0] tri-state (read cycle)	5	25	5	10	ns

- #1. If the S1D13505 host interface is disabled, the timing for WAIT# driven low is relative to the falling edge of RD0#, RD1#, WE0#, WE1# or the first positive edge of CLK after A[20:0], M/R# becomes valid, whichever one is later.
- #2. If the S1D13505 host interface is disabled, the timing for D[15:0] driven is relative to the falling edge of RD0#, RD1# or the first positive edge of CLK after A[20:0], M/R# becomes valid, whichever one is later.

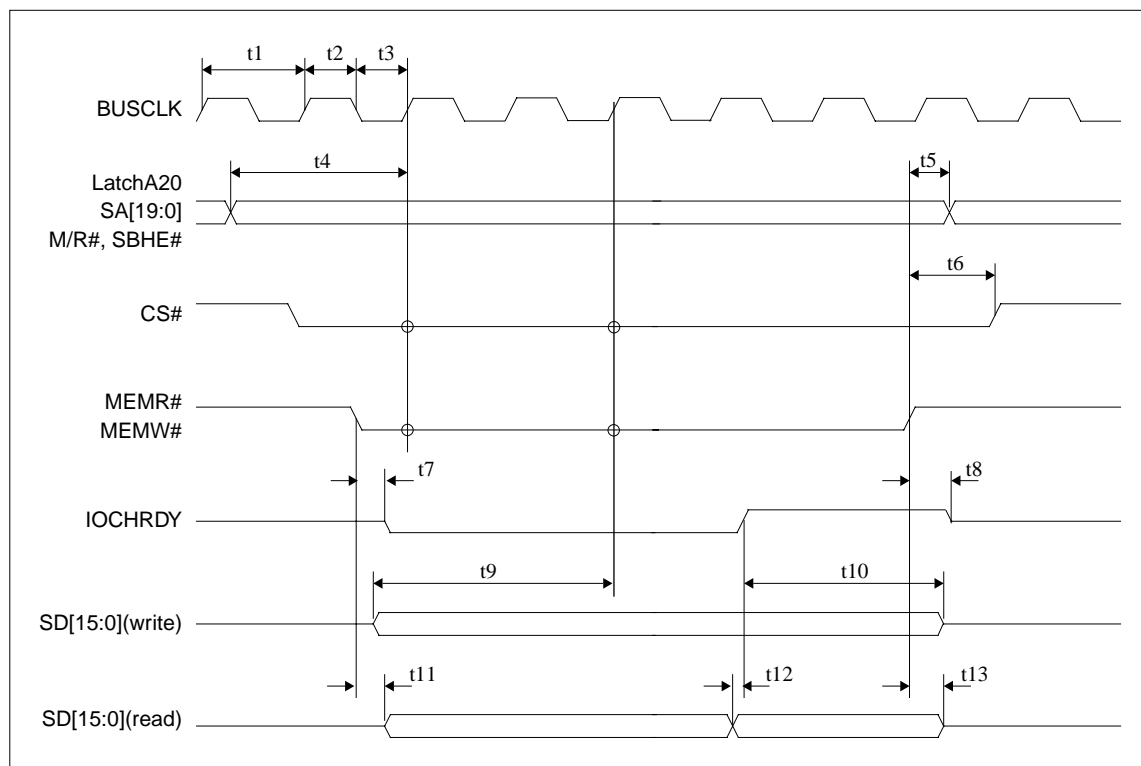
MIPS/ISA Interface Timing

Figure 7-7 MIPS/ISA Timing

Note: The above timing diagram is not applicable if the BUSCLK divided by 2 configuration option is selected.

Table 7-7 MIPS/ISA Timing

Symbol	Parameter	3.0V		5.0V		Units
		Min.	Max.	Min.	Max.	
t1	Clock period	20		20		ns
t2	Clock pulse width high	6		6		ns
t3	Clock pulse width low	6		6		ns
t4	LatchA20, SA[19:0], M/R#, SBHE# setup to first BUSCLK where CS# = 0 and either MEMR# = 0 or MEMW# = 0	10		10		ns
t5	LatchA20, SA[19:0], M/R#, SBHE# hold from rising edge of either MEMR# or MEMW#	0		0		ns
t6	CS# hold from rising edge of either MEMR# or MEMW#	0		0		ns
t7 ^{#1}	Falling edge of either MEMR# or MEMW# to IOCHRDY# driven low	0		0		ns
t8	Rising edge of either MEMR# or MEMW# to IOCHRDY# tri-state	5	25	2.5	10	ns
t9	SD[15:0] setup to third BUSCLK where CS# = 0 MEMW# = 0 (write cycle)	10		10		ns
t10	SD[15:0] hold (write cycle)	0		0		ns
t11 ^{#2}	Falling edge MEMR# toSD[15:0] driven (read cycle)	0		0		ns
t12	SD[15:0] setup to rising edge IOCHRDY# (read cycle)	0		0		ns
t13	Rising edge of MEMR# toSD[15:0] tri-state (read cycle)	5	25	5	10	ns

- #1. If the S1D13505 host interface is disabled, the timing for IOCHRDY driven low is relative to the falling edge of MEMR#, MEMW# or the first positive edge of BUSCLK after LatchA20, SA[19:0], M/R# becomes valid, whichever one is later.
- #2. If the S1D13505 host interface is disabled, the timing for SD[15:0] driven is relative to the falling edge of MEMR# or the first positive edge of BUSCLK after LatchA20, SA[19:0], M/R# becomes valid, whichever one is later.

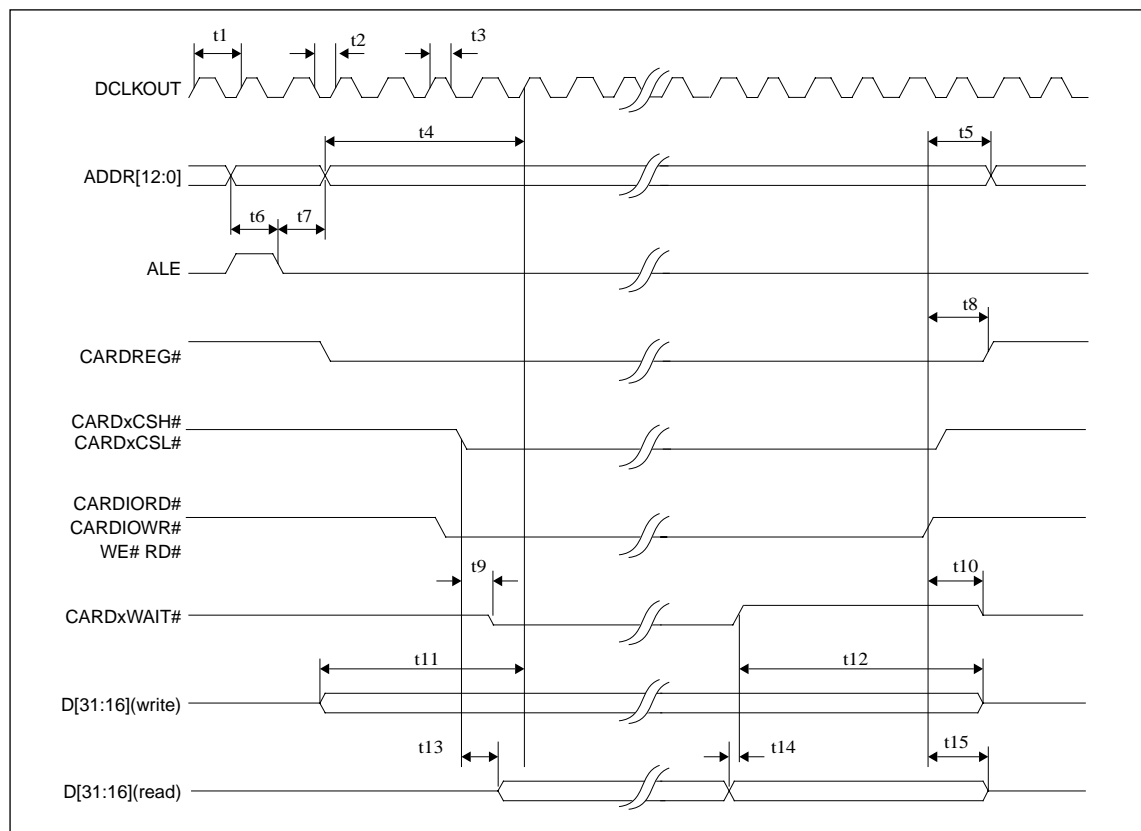
Philips Interface Timing (e.g. PR31500/PR31700)

Figure 7-8 Philips Timing

Table 7-8 Philips Timing

Symbol	Parameter	3.0V		5.0V		Units
		Min.	Max.	Min.	Max.	
t1	Clock period	13.3		13.3		ns
t2	Clock pulse width low	6		6		ns
t3	Clock pulse width high	6		6		ns
t4	ADDR[12:0] setup to first CLK of cycle	10		10		ns
t5	ADDR[12:0] hold from command invalid	0		0		ns
t6	ADDR[12:0] setup to falling edge ALE	10		10		ns
t7	ADDR[12:0] hold from falling edge ALE	5		5		ns
t8	CARDREG# hold from command invalid	0		0		ns
t9#1	Falling edge of chip select to CARDxWAIT# driven	0	15	0	9	ns
t10	Command invalid to CARDxWAIT# tri-state	5	25	2.5	10	ns
t11	D[31:16] valid to first CLK of cycle (write cycle)	10		10		ns
t12	D[31:16] hold from rising edge of CARDxWAIT#	0		0		ns
t13#2	Chip select to D[31:16] driven (read cycle)	1		1		ns
t14	D[31:16] setup to rising edge CARDxWAIT# (read cycle)	0		0		ns
t15	Command invalid to D[31:16] tri-state (read cycle)	5	25	2.5	10	ns

- #1. If the S1D13505 host interface is disabled, the timing for CARDxWAIT# driven is relative to the falling edge of chip select or the second positive edge of DCLKOUT after ADDR[12:0] becomes valid, whichever one is later.
- #2. If the S1D13505 host interface is disabled, the timing for D[31:16] driven is relative to the falling edge of chip select or the second positive edge of DCLKOUT after ADDR[12:0] becomes valid, whichever one is later.

Note: The Philips interface has different clock input requirements as follows:

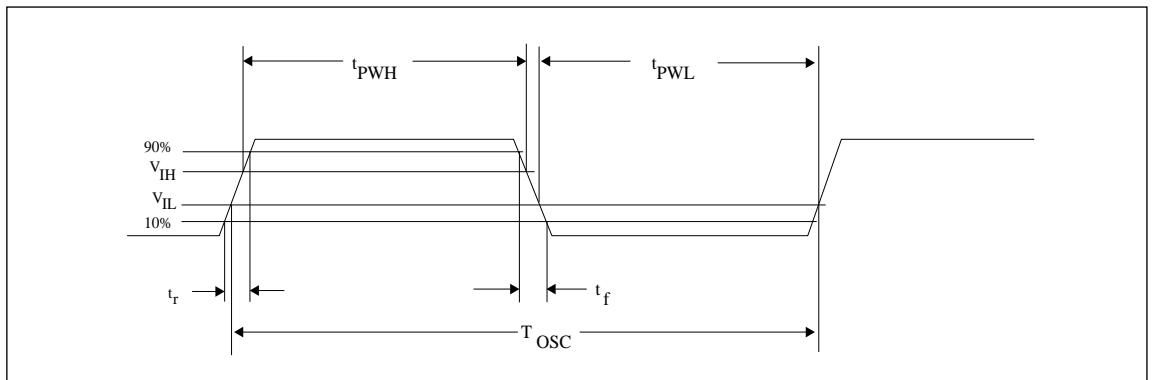


Figure 7-9 Clock Input Requirements for BUSCLK Using Philips Local Bus

Table 7-9 Clock Input Requirements for BUSCLK Using Philips Local Bus

Symbol	Parameter	Min.	Max.	Units
Tosc	Input Clock Period	13.3		ns
tpWH	Input Clock Pulse Width High	6		ns
tpWL	Input Clock Pulse Width Low	6		ns
tr	Input Clock Fall Time (10%–90%)		5	ns
tr	Input Clock Rise Time (10%–90%)		5	ns

Toshiba Interface Timing (e.g. TX3912)

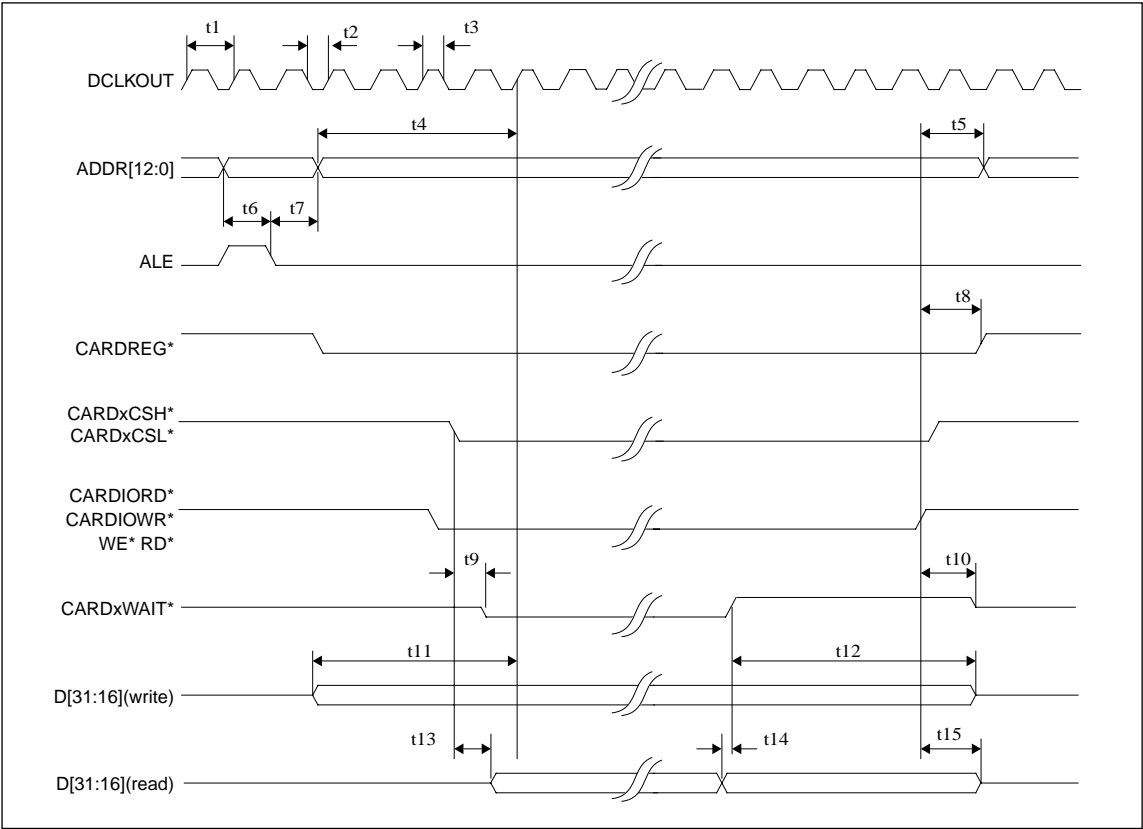


Figure 7-10 Toshiba Timing

Table 7-10 Toshiba Timing

Symbol	Parameter	3.0V		5.0V		Units
		Min.	Max.	Min.	Max.	
t1	Clock period	13.3		13.3		ns
t2	Clock pulse width low	5.4		5.4		ns
t3	Clock pulse width high	5.4		5.4		ns
t4	ADDR[12:0] setup to first CLK of cycle	10		10		ns
t5	ADDR[12:0] hold from command invalid	0		0		ns
t6	ADDR[12:0] setup to falling edge ALE	10		10		ns
t7	ADDR[12:0] hold from falling edge ALE	5		5		ns
t8	CARDREG* hold from command invalid	0		0		ns
t9#1	Falling edge of chip select to CARDxWAIT* driven	0	15	0	9	ns
t10	Command invalid to CARDxWAIT* tri-state	5	25	2.5	10	ns
t11	D[31:16] valid to first CLK of cycle (write cycle)	10		10		ns
t12	D[31:16] hold from rising edge of CARDxWAIT*	0		0		ns
t13#2	Chip select to D[31:16] driven (read cycle)	1		1		ns
t14	D[31:16] setup to rising edge CARDxWAIT* (read cycle)	0		0		ns
t15	Command invalid to D[31:16] tri-state (read cycle)	5	25	2.5	10	ns

- #1. If the S1D13505 host interface is disabled, the timing for CARDxWAIT* driven is relative to the falling edge of chip select or the second positive edge of DCLKOUT after ADDR[12:0] becomes valid, whichever one is later.
- #2. If the S1D13505 host interface is disabled, the timing for D[31:16] driven is relative to the falling edge of chip select or the second positive edge of DCLKOUT after ADDR[12:0] becomes valid, whichever one is later.

Note: The Toshiba interface has different clock input requirements as follows:

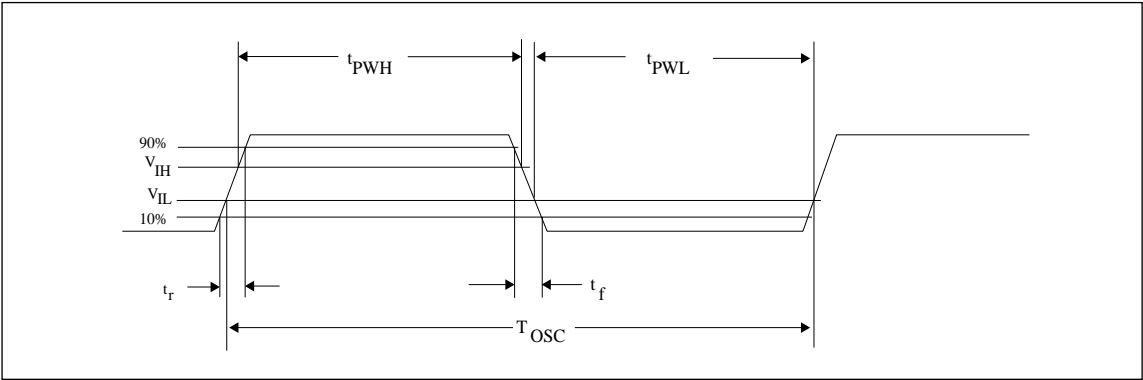


Figure 7-11 Clock Input Requirements

Table 7-11 Clock Input Requirements for BUSCLK Using Toshiba Local Bus

Symbol	Parameter	Min.	Max.	Units
Tosc	Input Clock Period	13.3		ns
tPWH	Input Clock Pulse Width High	5.4		ns
tPWL	Input Clock Pulse Width Low	5.4		ns
tf	Input Clock Fall Time (10%–90%)		5	ns
tr	Input Clock Rise Time (10%–90%)		5	ns

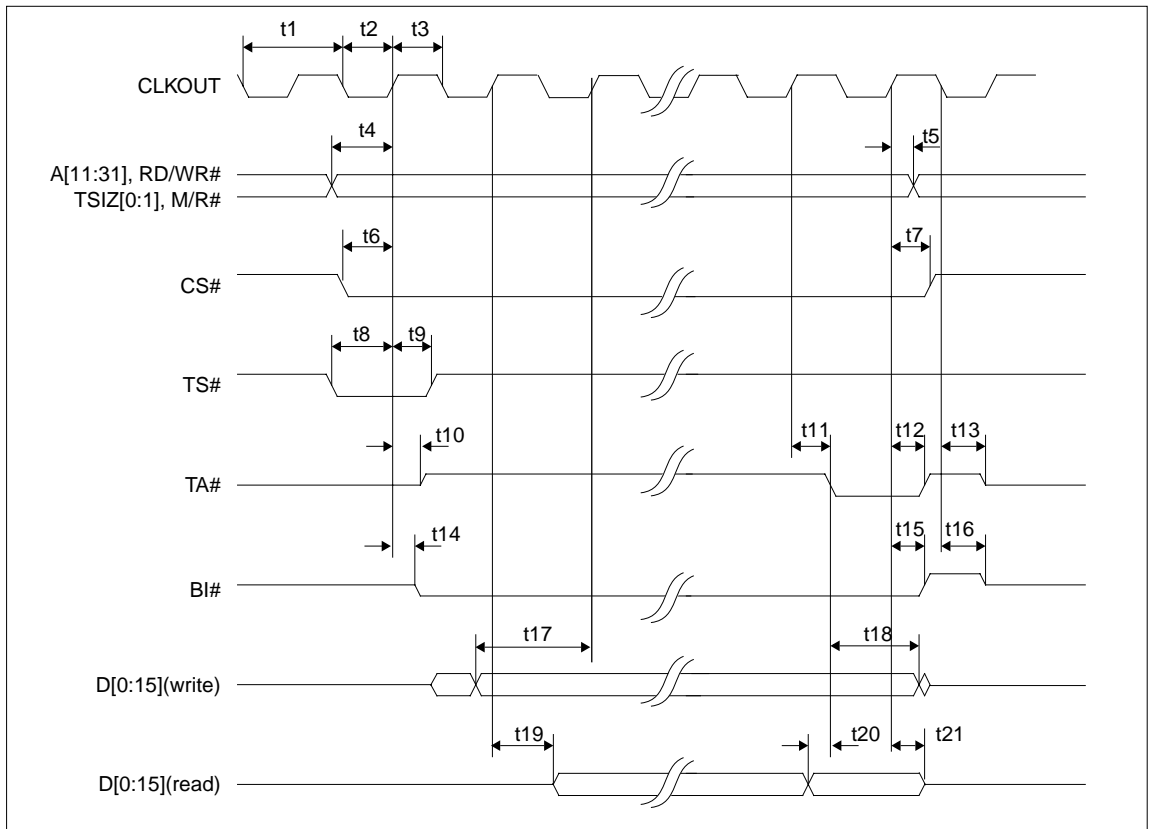
PowerPC Interface Timing (e.g. MPC8xx, MC68040, Coldfire)

Figure 7-12 PowerPC Timing

Note: The above timing diagram is not applicable if the BUSCLK divided by 2 configuration option is selected.

Table 7-12 PowerPC Timing

Symbol	Parameter	3.0V		5.0V		Units
		Min.	Max.	Min.	Max.	
t_1	Clock period	25		20		ns
t_2	Clock pulse width high	6		6		ns
t_3	Clock pulse width low	6		6		ns
t_4	AB[11:31], RD/WR#, TSIZ[0:1], M/R# setup	10		10		ns
t_5	AB[11:31], RD/WR#, TSIZ[0:1], M/R# hold	0		0		ns
t_6	CS# setup	10		10		ns
t_7	CS# hold	0		0		ns
t_8	TS# setup	7		10		ns
t_9	TS# hold	5		0		ns
t_{10}	CLKOUT to TA# driven	0		0		ns
t_{11}	CLKOUT to TA# low	3	19	3	12	ns
t_{12}	CLKOUT to TA# high	3	19.7	3	13	ns
t_{13}	negative edge CLKOUT to TA# tri-state	5	25	2.5	10	ns
t_{14}	CLKOUT to BI# driven	0	18	0	11	ns
t_{15}	CLKOUT to BI# high	3	16	3	10	ns
t_{16}	negative edge CLKOUT to BI# tri-state	5	25	2.5	10	ns
t_{17}	DB[15:0] setup to 2nd CLKOUT after TS# = 0 (write cycle)	10		10		ns
t_{18}	DB[15:0] hold (write cycle)	0		0		ns
t_{19}	CLKOUT to DB driven (read cycle)	0		0		ns
t_{20}	DB[15:0] valid to TA# falling edge (read cycle)	0		0		ns
t_{21}	CLKOUT to DB[15:0] tri-state (read cycle)	5	25	2.5	10	ns

7.2 Clock Input Requirements

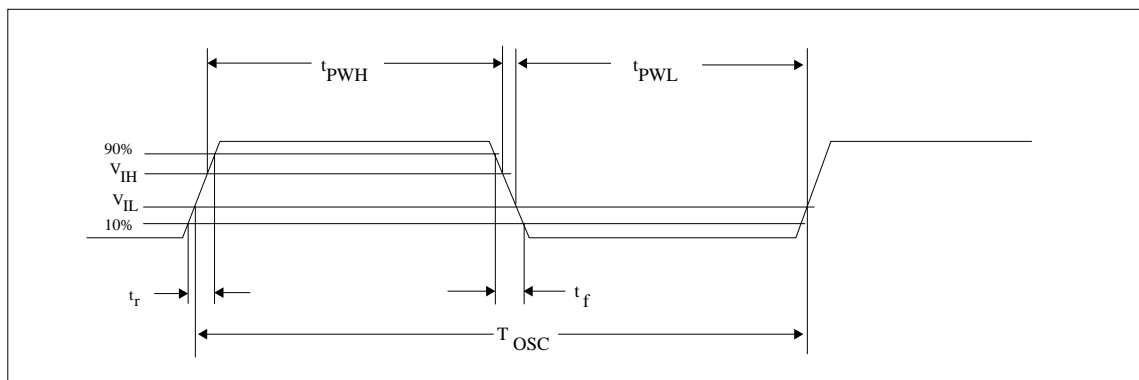


Figure 7-13 Clock Input Requirements

Table 7-13 Clock Input Requirements for CLKI Divided Down Internally (MCLK = CLKI/2)

Symbol	Parameter	Min.	Max.	Units
TOSC	Input Clock Period	12.5		ns
tPWH	Input Clock Pulse Width High	5.6		ns
tPWL	Input Clock Pulse Width Low	5.6		ns
t _f	Input Clock Fall Time (10% - 90%)		5	ns
t _r	Input Clock Rise Time (10% - 90%)		5	ns

Table 7-14 Clock Input Requirements for CLKI

Symbol	Parameter	Min.	Max.	Units
TOSC	Input Clock Period	25		ns
tPWH	Input Clock Pulse Width High	11.3		ns
tPWL	Input Clock Pulse Width Low	11.3		ns
t _f	Input Clock Fall Time (10% - 90%)		5	ns
t _r	Input Clock Rise Time (10% - 90%)		5	ns

Note: When CLKI is more than 40MHz, REG[19h] bit 2 must be set to 1 (MCLK = CLKI/2).

7.3 Memory Interface Timing

EDO-DRAM Read/Write/Read-Write Timing

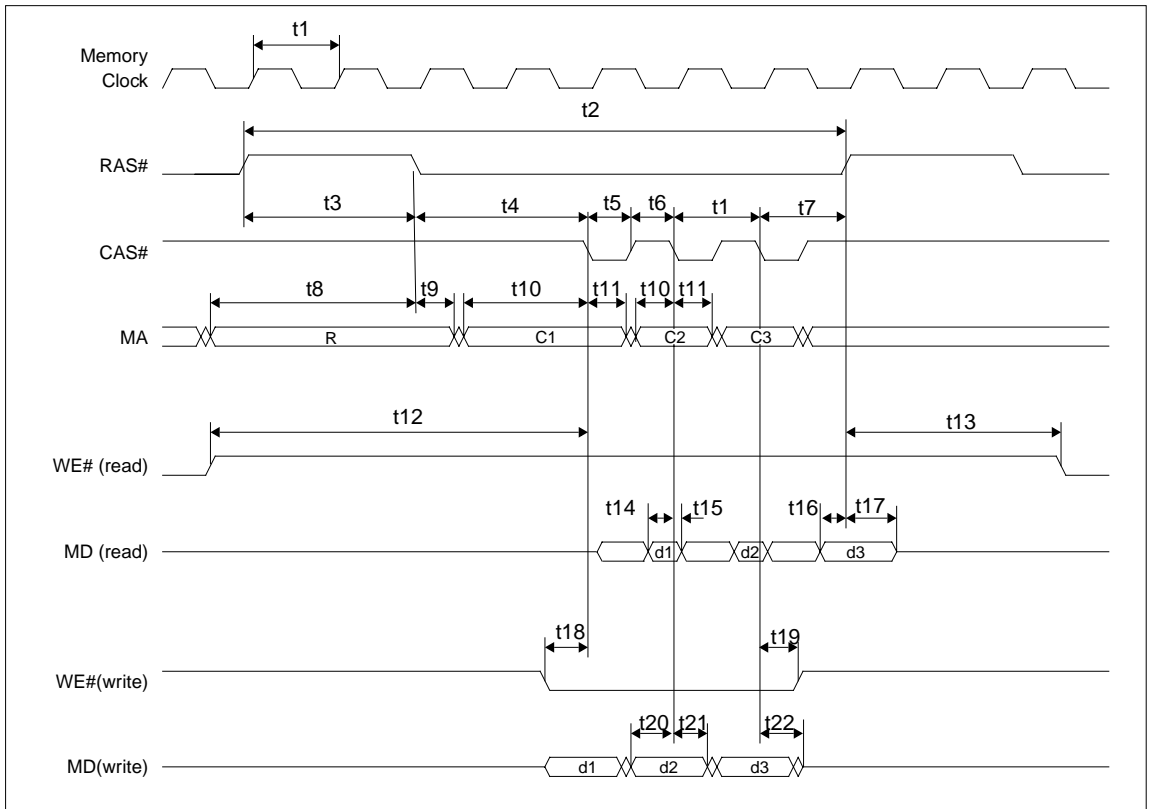


Figure 7-14 EDO-DRAM Read/Write Timing

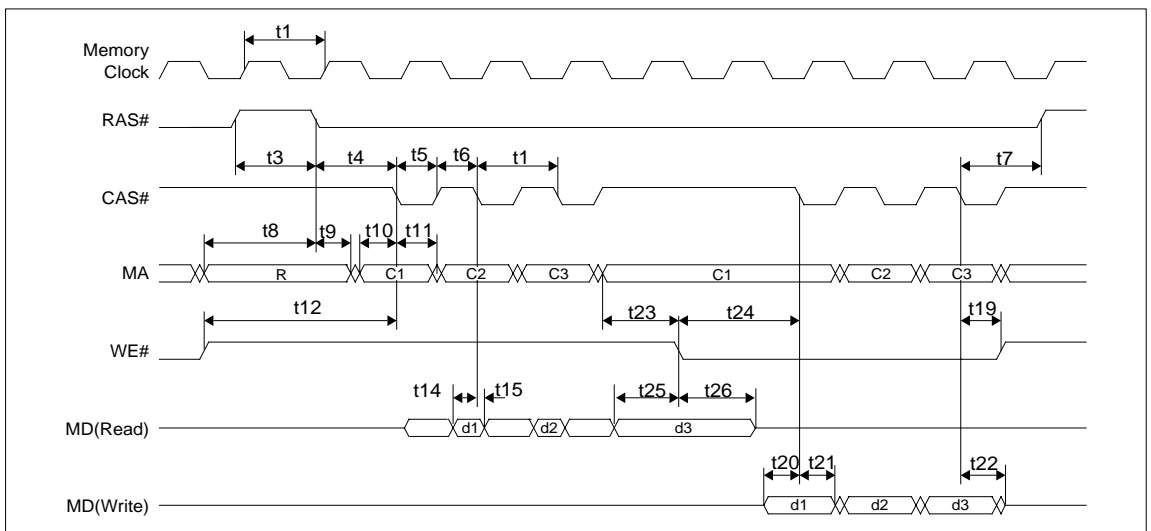


Figure 7-15 EDO-DRAM Read-Write Timing

Table 7-15 EDO DRAM Read Timing

Symbol	Parameter	Min.	Max.	Units
t1	Internal memory clock period	25		ns
t2	Random read cycle REG[22h] bits 6-5 = 00	5 t1		ns
	Random read cycle REG[22h] bits 6-5 = 01	4 t1		ns
	Random read cycle REG[22h] bits 6-5 = 10	3 t1		ns
t3	RAS# precharge time (REG[22h] bits 3-2 = 00)	2 t1 - 3		ns
	RAS# precharge time (REG[22h] bits 3-2 = 01)	1.45 t1 - 3		ns
	RAS# precharge time (REG[22h] bits 3-2 = 10)	1 t1 - 3		ns
t4	RAS# to CAS# delay time (REG[22h] bit 4 = 0 and bits 3-2 = 00 or 10)	2 t1 - 3		ns
	RAS# to CAS# delay time (REG[22h] bit 4 = 1 and bits 3-2 = 00 or 10)	1 t1 - 3		ns
	RAS# to CAS# delay time (REG[22h] bits 3-2 = 01)	1.45 t1 - 3		ns
t5	CAS# precharge time	0.45 t1 - 3		ns
t6	CAS# pulse width	0.45 t1 - 3		ns
t7	RAS# hold time	1 t1 - 3		ns
t8	Row address setup time (REG[22h] bits 3-2 = 00)	2.45 t1		ns
	Row address setup time (REG[22h] bits 3-2 = 01)	2 t1		ns
	Row address setup time (REG[22h] bits 3-2 = 10)	1.45 t1		ns
t9	Row address hold time (REG[22h] bits 3-2 = 00 or 10)	0.45 t1 - 3		ns
	Row address hold time (REG[22h] bits 3-2 = 01)	1 t1 - 3		ns
t10	Column address setup time	0.45 t1 - 3		ns
t11	Column address hold time	0.45 t1 - 3		ns
t12	Read Command Setup (REG[22h] bit 4 = 0 and bits 3-2 = 00)	4.45 t1 - 3		ns
	Read Command Setup (REG[22h] bit 4 = 0 and bits 3-2 = 10)	3.45 t1 - 3		ns
	Read Command Setup (REG[22h] bit 4 = 1 and bits 3-2 = 00)	3.45 t1 - 3		ns
	Read Command Setup (REG[22h] bit 4 = 1 and bits 3-2 = 10)	2.45 t1 - 3		ns
	Read Command Setup (REG[22h] bits 3-2 = 01)	3.45 t1 - 3		ns
t13	Read Command Hold (REG[22h] bit 4 = 0 and bits 3-2 = 00)	3.45 t1 - 3		ns
	Read Command Hold (REG[22h] bit 4 = 0 and bits 3-2 = 10)	2.45 t1 - 3		ns
	Read Command Hold (REG[22h] bit 4 = 1 and bits 3-2 = 00)	2.45 t1 - 3		ns
	Read Command Hold (REG[22h] bit 4 = 1 and bits 3-2 = 10)	1.45 t1 - 3		ns
	Read Command Hold (REG[22h] bits 3-2 = 01)	2.45 t1 - 3		ns
t14	Read Data Setup referenced from CAS#	5		ns
t15	Read Data Hold referenced from CAS#	3		ns
t16	Last Read Data Setup referenced from RAS#	5		ns
t17	Bus Turn Off from RAS#	3	t1 - 5	ns
t18	Write Command Setup	0.45 t1 - 3		ns
t19	Write Command Hold	0.45 t1 - 3		ns
t20	Write Data Setup	0.45 t1 - 3		ns
t21	Write Data Hold	0.45 t1 - 3		ns
t22	MD Tri-state	0.45 t1	0.45 t1 + 21	ns
t23	CAS# to WE# active during Read-Write cycle	1 t1 - 3		ns
t24	Write Command Setup during Read-Write cycle	1.45 t1 - 3		ns
t25	Last Read Data Setup referenced from WE# during Read-Write cycle	10		ns
t26	Bus Tri-state from WE# during Read-Write cycle	0	t1 - 5	ns

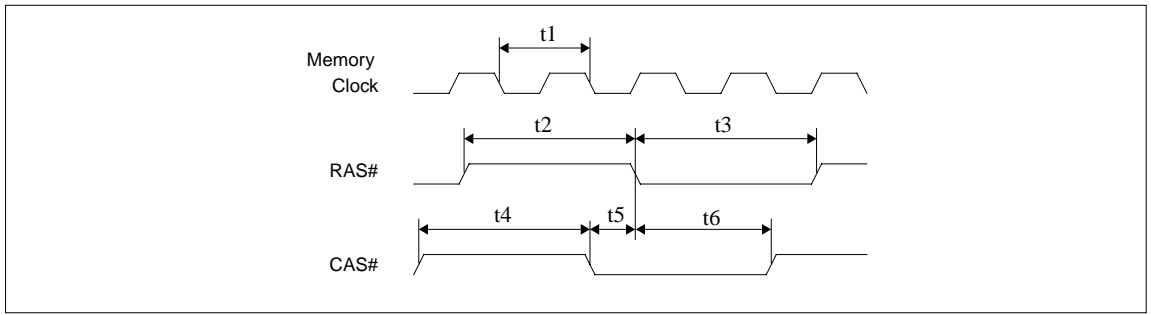
EDO-DRAM CAS Before RAS Refresh Timing

Figure 7-16 EDO-DRAM CAS Before RAS Refresh Write Timing

Table 7-16 EDO DRAM CAS Before RAS Refresh Write Timing

Symbol	Parameter	Min.	Max.	Units
t1	Internal memory clock period	25		ns
t2	RAS# precharge time (REG[22h] bits 3-2 = 00)	2 t1 - 3		ns
	RAS# precharge time (REG[22h] bits 3-2 = 01)	1.45 t1 - 3		ns
	RAS# precharge time (REG[22h] bits 3-2 = 10)	1 t1 - 3		ns
t3	RAS# pulse width (REG[22h] bits 6-5 = 00 and bits 3-2 = 00)	3 t1 - 3		ns
	RAS# pulse width (REG[22h] bits 6-5 = 00 and bits 3-2 = 01)	3.45 t1 - 3		ns
	RAS# pulse width (REG[22h] bits 6-5 = 00 and bits 3-2 = 10)	4 t1 - 3		ns
	RAS# pulse width (REG[22h] bits 6-5 = 01 and bits 3-2 = 00)	2 t1 - 3		ns
	RAS# pulse width (REG[22h] bits 6-5 = 01 and bits 3-2 = 01)	2.45 t1 - 3		ns
	RAS# pulse width (REG[22h] bits 6-5 = 01 and bits 3-2 = 10)	3 t1 - 3		ns
	RAS# pulse width (REG[22h] bits 6-5 = 10 and bits 3-2 = 00)	1 t1 - 3		ns
	RAS# pulse width (REG[22h] bits 6-5 = 10 and bits 3-2 = 01)	1.45 t1 - 3		ns
	RAS# pulse width (REG[22h] bits 6-5 = 10 and bits 3-2 = 10)	2 t1 - 3		ns
t4	CAS# pulse width	t2		ns
t5	CAS# setup time (REG[22h] bits 3-2 = 00 or 10)	0.45 t1 - 3		ns
	CAS# setup time (REG[22h] bits 3-2 = 01)	1 t1 - 3		ns
t6	CAS# Hold to RAS# (REG[22h] bits 6-5 = 00 and bits 3-2 = 00)	2.45 t1 - 3		ns
	CAS# Hold to RAS# (REG[22h] bits 6-5 = 00 and bits 3-2 = 01)	3 t1 - 3		ns
	CAS# Hold to RAS# (REG[22h] bits 6-5 = 00 and bits 3-2 = 10)	3.45 t1 - 3		ns
	CAS# Hold to RAS# (REG[22h] bits 6-5 = 01 and bits 3-2 = 00)	1.45 t1 - 3		ns
	CAS# Hold to RAS# (REG[22h] bits 6-5 = 01 and bits 3-2 = 01)	2 t1 - 3		ns
	CAS# Hold to RAS# (REG[22h] bits 6-5 = 01 and bits 3-2 = 10)	2.45 t1 - 3		ns
	CAS# Hold to RAS# (REG[22h] bits 6-5 = 10 and bits 3-2 = 00)	0.45 t1 - 3		ns
	CAS# Hold to RAS# (REG[22h] bits 6-5 = 10 and bits 3-2 = 01)	1 t1 - 3		ns
	CAS# Hold to RAS# (REG[22h] bits 6-5 = 10 and bits 3-2 = 10)	1.45 t1 - 3		ns

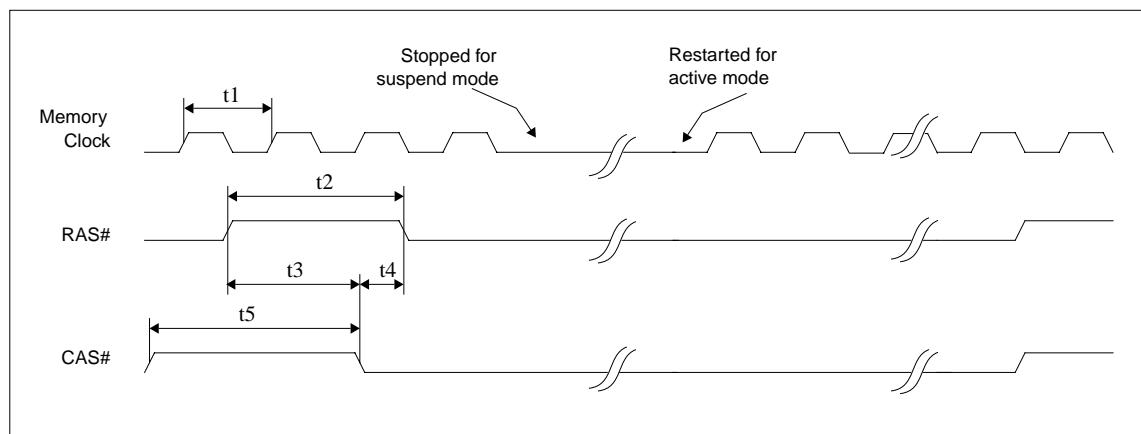
EDO-DRAM Self-Refresh Timing

Figure 7-17 EDO-DRAM Self-Refresh Timing

Table 7-17 EDO-DRAM Self-Refresh Timing

Symbol	Parameter	Min.	Max.	Units
t_1	Internal memory clock period	25		ns
t_2	RAS# precharge time (REG[22h] bits 3-2 = 00)	2 t_1 - 3		ns
	RAS# precharge time (REG[22h] bits 3-2 = 01)	1.45 t_1 - 3		ns
	RAS# precharge time (REG[22h] bits 3-2 = 10)	1 t_1 - 3		ns
t_3	RAS# to CAS# precharge time (REG[22h] bits 3-2 = 00)	1.45 t_1 - 3		ns
	RAS# to CAS# precharge time (REG[22h] bits 3-2 = 01 or 10)	0.45 t_1 - 3		ns
t_4	CAS# setup time (REG[22h] bits 3-2 = 00 or 10)	0.45 t_1 - 3		ns
	CAS# setup time (REG[22h] bits 3-2 = 01)	1 t_1 - 3		ns
t_5	CAS# precharge time (REG[22h] bits 3-2 = 00)	2 t_1 - 3		ns
	CAS# precharge time (REG[22h] bits 3-2 = 01 or 10)	1 t_1 - 3		ns

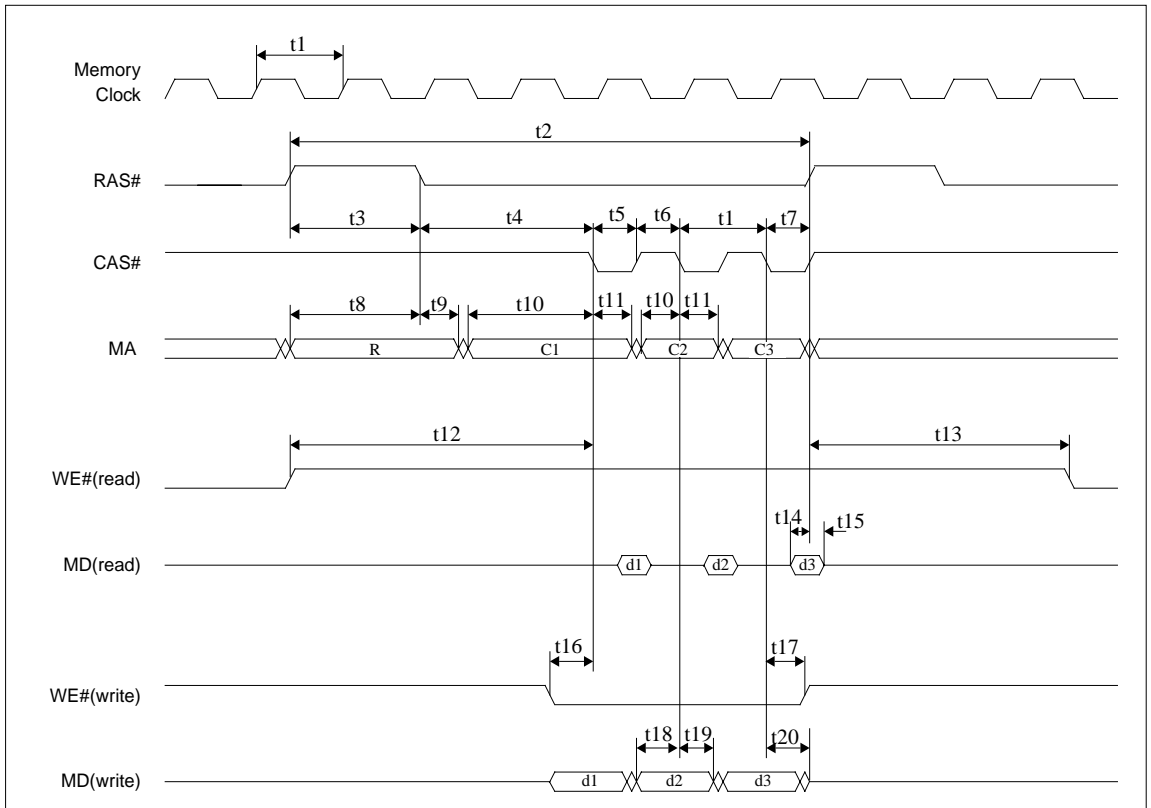
FPM-DRAM Read / Write / Read - Write Timing

Figure 7-18 FPM-DRAM Read/Write Timing

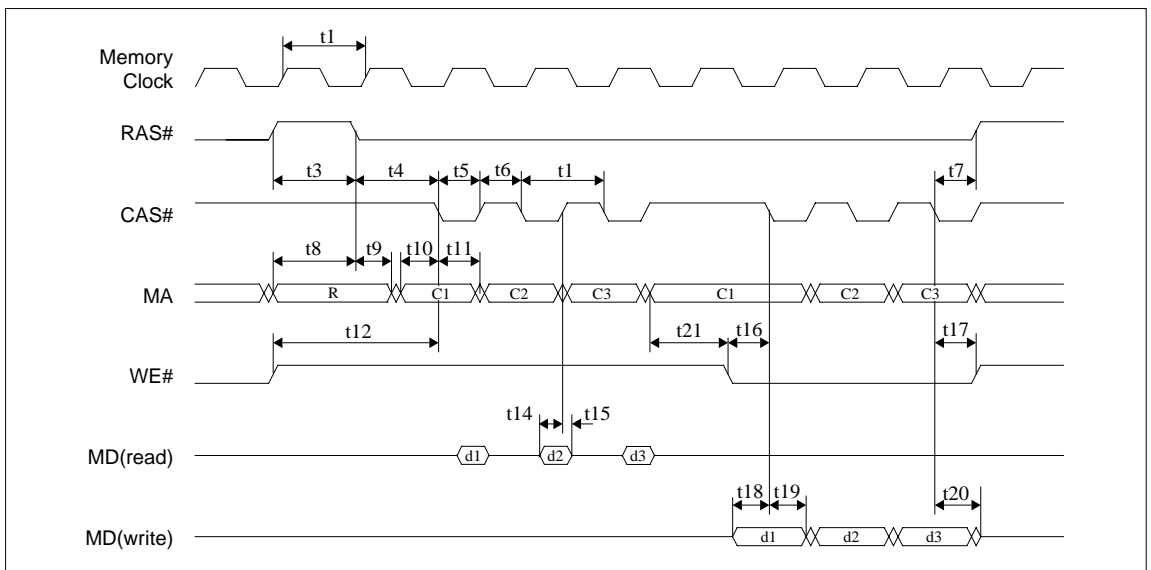


Figure 7-19 FPM-DRAM Read-Write Timing

Table 7-18 FPM-DRAM Read/Write/Read-Write Timing

Symbol	Parameter	Min.	Max.	Units
t1	Internal memory clock period	40		ns
t2	Random read cycle REG[22h] bits 6-5 = 00	5 t1		ns
	Random read cycle REG[22h] bits 6-5 = 01	4 t1		ns
	Random read cycle REG[22h] bits 6-5 = 10	3 t1		ns
t3	RAS# precharge time (REG[22h] bits 3-2 = 00)	2 t1 - 3		ns
	RAS# precharge time (REG[22h] bits 3-2 = 01)	1.45 t1 - 3		ns
	RAS# precharge time (REG[22h] bits 3-2 = 10)	1 t1 - 3		ns
t4	RAS# to CAS# delay time (REG[22h] bit 4 = 1 and bits 3-2 = 00 or 10)	1.45 t1 - 3		ns
	RAS# to CAS# delay time (REG[22h] bit 4 = 0 and bits 3-2 = 00 or 10)	2.45 t1 - 3		ns
	RAS# to CAS# delay time (REG[22h] bit 4 = 1 and bits 3-2 = 01)	1 t1 - 3		ns
	RAS# to CAS# delay time (REG[22h] bit 4 = 0 and bits 3-2 = 01)	2 t1 - 3		ns
t5	CAS# precharge time	0.45 t1 - 3		ns
t6	CAS# pulse width	0.45 t1 - 3		ns
t7	RAS# hold time	0.45 t1 - 3		ns
t8	Row address setup time (REG[22h] bits 3-2 = 00)	2 t1 - 3		ns
	Row address setup time (REG[22h] bits 3-2 = 01)	1.45 t1 - 3		ns
	Row address setup time (REG[22h] bits 3-2 = 10)	1 t1 - 3		ns
t9	Row address hold time (REG[22h] bits 3-2 = 00 or 10)	t1 - 3		ns
	Row address hold time (REG[22h] bits 3-2 = 01)	0.45 t1 - 3		ns
t10	Column address setup time	0.45 t1 - 3		ns
t11	Column address hold time	0.45 t1 - 3		ns
t12	Read Command Setup (REG[22h] bit 4 = 0 and bits 3-2 = 00)	4.45 t1 - 3		ns
	Read Command Setup (REG[22h] bit 4 = 0 and bits 3-2 = 01 or 10)	3.45 t1 - 3		ns
	Read Command Setup (REG[22h] bit 4 = 1 and bits 3-2 = 00)	3.45 t1 - 3		ns
	Read Command Setup (REG[22h] bit 4 = 1 and bits 3-2 = 01 or 10)	2.45 t1 - 3		ns
t13	Read Command Hold (REG[22h] bit 4 = 0 and bits 3-2 = 00)	4 t1 - 3		ns
	Read Command Hold (REG[22h] bit 4 = 0 and bits 3-2 = 01 or 10)	3 t1 - 3		ns
	Read Command Hold (REG[22h] bit 4 = 1 and bits 3-2 = 00)	3 t1 - 3		ns
	Read Command Hold (REG[22h] bit 4 = 1 and bits 3-2 = 01 or 10)	2 t1 - 3		ns
t14	Read Data Setup referenced from CAS#	5		ns
t15	Bus Tri-State	3	t1 - 5	ns
t16	Write Command Setup	0.45 t1 - 3		ns
t17	Write Command Hold	0.45 t1 - 3		ns
t18	Write Data Setup	0.45 t1 - 3		ns
t19	Write Data Hold	0.45 t1 - 3		ns
t20	MD Tri-state	0.45 t1	0.45 t1 + 21	ns
t21	CAS# to WE# active during Read-Write cycle	0.45 t1 - 3		ns

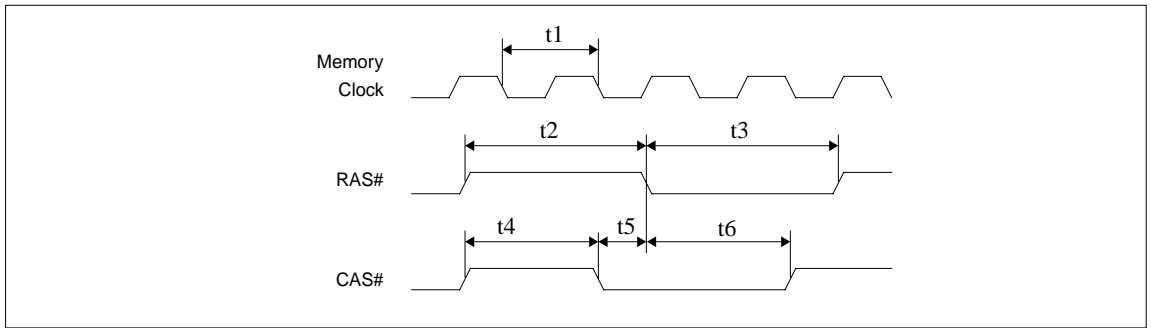
FPM-DRAM CAS Before RAS Refresh Timing

Figure 7-20 FPM-DRAM CAS before RAS Refresh Timing

Table 7-19 FPM-DRAM CAS before RAS Refresh Timing

Symbol	Parameter	Min.	Max.	Units
t1	Internal memory clock period	40		ns
t2	RAS# precharge time (REG[22h] bits 3-2 = 00)	2.45 t1 - 3		ns
	RAS# precharge time (REG[22h] bits 3-2 = 01 or 10)	1.45 t1 - 3		ns
t3	RAS# pulse width (REG[22h] bits 6-5 = 00 and bits 3-2 = 00)	2.45 t1 - 3		ns
	RAS# pulse width (REG[22h] bits 6-5 = 00 and bits 3-2 = 01 or 10)	3.45 t1 - 3		ns
	RAS# pulse width (REG[22h] bits 6-5 = 01 and bits 3-2 = 00)	1.45 t1 - 3		ns
	RAS# pulse width (REG[22h] bits 6-5 = 01 and bits 3-2 = 01 or 10)	2.45 t1 - 3		ns
	RAS# pulse width (REG[22h] bits 6-5 = 10 and bits 3-2 = 00)	0.45 t1 - 3		ns
	RAS# pulse width (REG[22h] bits 6-5 = 10 and bits 3-2 = 01 or 10)	1.45 t1 - 3		ns
t4	CAS# pulse width (REG[22h] bits 3-2 = 00)	2 t1 - 3		ns
	CAS# pulse width (REG[22h] bits 3-2 = 01 or 10)	1 t1 - 3		ns
t5	CAS# Setup to RAS#	0.45 t1 - 3		ns
t6	CAS# Hold to RAS# (REG[22h] bits 6-5 = 00 and bits 3-2 = 00)	2.45 t1 - 3		ns
	CAS# Hold to RAS# (REG[22h] bits 6-5 = 00 and bits 3-2 = 01 or 10)	3.45 t1 - 3		ns
	CAS# Hold to RAS# (REG[22h] bits 6-5 = 01 and bits 3-2 = 00)	1.45 t1 - 3		ns
	CAS# Hold to RAS# (REG[22h] bits 6-5 = 01 and bits 3-2 = 01 or 10)	2.45 t1 - 3		ns
	CAS# Hold to RAS# (REG[22h] bits 6-5 = 10 and bits 3-2 = 00)	0.45 t1 - 3		ns
	CAS# Hold to RAS# (REG[22h] bits 6-5 = 10 and bits 3-2 = 01 or 10)	1.45 t1 - 3		ns

FPM-DRAM Self-Refresh Timing

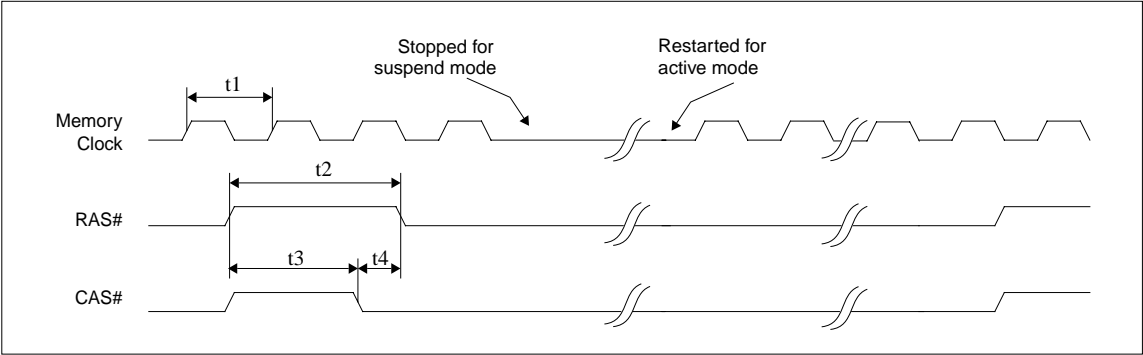


Figure 7-21 FPM-DRAM Self-Refresh Timing

Table 7-20 FPM DRAM Self-Refresh Timing

Symbol	Parameter	Min.	Max.	Units
t1	Internal memory clock	40		ns
t2	RAS# precharge time (REG[22h] bits 3-2 = 00)	2.45 t1 - 1		ns
	RAS# precharge time (REG[22h] bits 3-2 = 01 or 10)	1.45 t1 - 1		ns
t3	RAS# to CAS# precharge time (REG[22h] bits 3-2 = 00)	2 t1		ns
	RAS# to CAS# precharge time (REG[22h] bits 3-2 = 01 or 10)	1 t1		ns
t4	CAS# setup time (CAS# before RAS# refresh)	0.45 t1 - 2		ns

7.4 Power Sequencing

LCD Power Sequencing

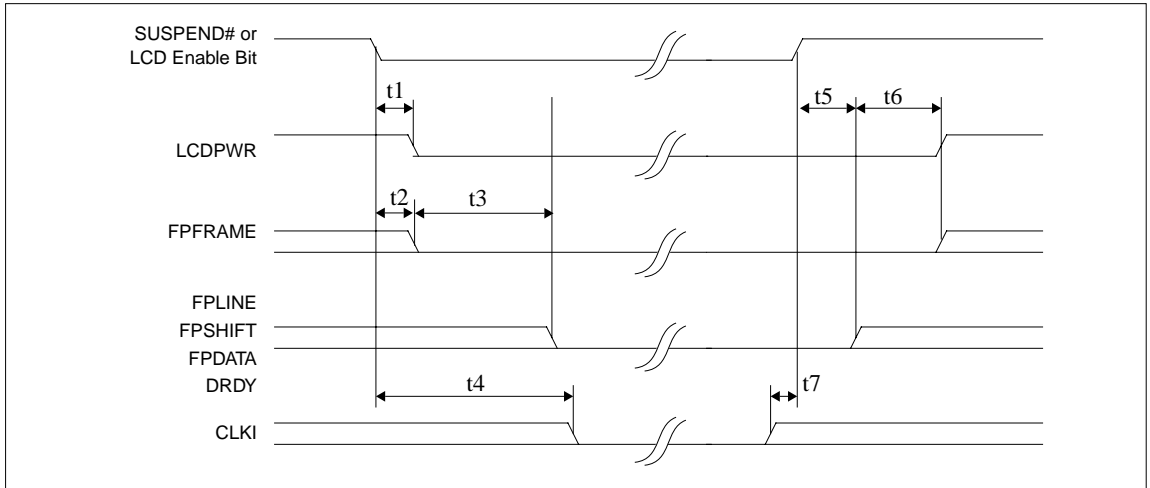


Figure 7-22 LCD Panel Power Off / Power On Timing. Drawn with LCDPWR Set to Active High Polarity

Table 7-21 LCD Panel Power Off/ Power On

Symbol	Parameter	Min.	Max.	Units
t1	SUSPEND# or LCD ENABLE BIT low to LCDPWR off		$2T_{FPFRAME} + 8T_{PCLK}$	ns
t2	SUSPEND# or LCD ENABLE BIT low to FPFAME inactive		1	Frames
t3	FPFRAME inactive to FPLINE, FPSHIFT, FPDATA, DRDY inactive	128		Frames
t4	SUSPEND# to CLKI inactive	130		Frames
t5	SUSPEND# or LCD ENABLE BIT high to FPLINE, FPSHIFT, FPDATA, DRDY active		$T_{FPFRAME} + 8T_{PCLK}$	ns
t6	FPLINE, FPSHIFT, FPDATA, DRDY active to LCDPWR, on and FPFAME active	128		Frames
t7	CLKI active to SUSPEND# inactive	0		ns

Note: Where $T_{FPFRAME}$ is the period of FPFAME and T_{PCLK} is the period of the pixel clock.

Power Save Status

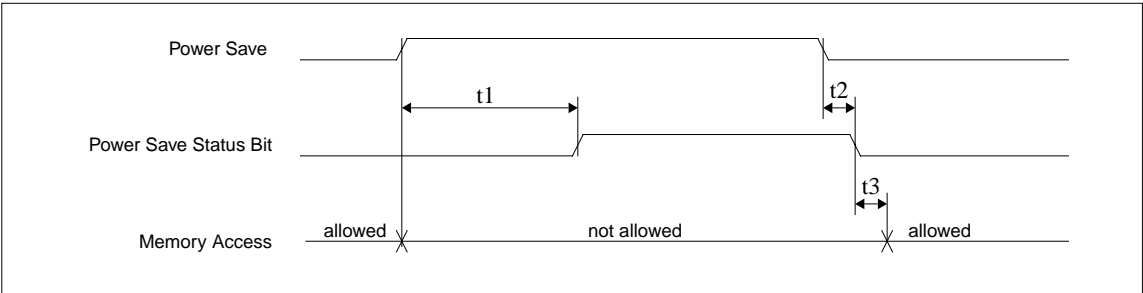


Figure 7-23 Power Save Status and Local Bus Memory Access Relative to Power Save Mode

Note: Power Save can be initiated through either the SUSPEND# pin or Software Suspend Enable Bit.

Table 7-22 Power Save Status and Local Bus Memory Access Relative to Power Save Mode

Symbol	Parameter	Min.	Max.	Units
t1	Power Save initiated to rising edge of Power Save Status and the last time memory access by the local bus may be performed	129	130	Frames
t2	Power Save deactivated to falling edge of Power Save Status		12	MCLK
t3	Falling edge of Power Save Status to the earliest time the local bus may perform a memory access		8	MCLK

Note: It is recommended that memory access not be performed after a Power Save Mode has been initiated.

7.5 Display Interface

4-Bit Single Monochrome Passive LCD Panel Timing

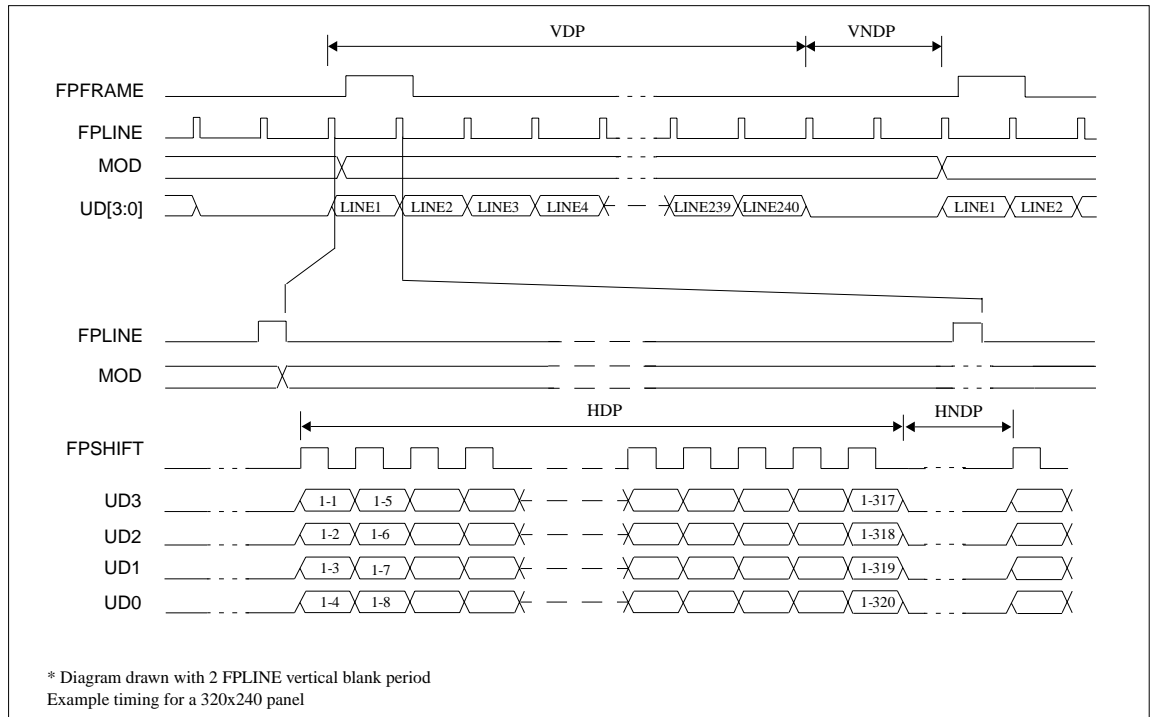


Figure 7-24 4-Bit Single Monochrome Passive LCD Panel Timing

VDP	= Vertical Display Period	= (REG[09h] bits [1:0], REG[08h] bits [7:0]) + 1
VNDP	= Vertical Non-Display Period	= (REG[0Ah] bits [5:0]) + 1
HDP	= Horizontal Display Period	= ((REG[04h] bits [6:0]) + 1) * 8Ts
HNDP	= Horizontal Non-Display Period	= ((REG[05h] bits [4:0]) + 1) * 8Ts

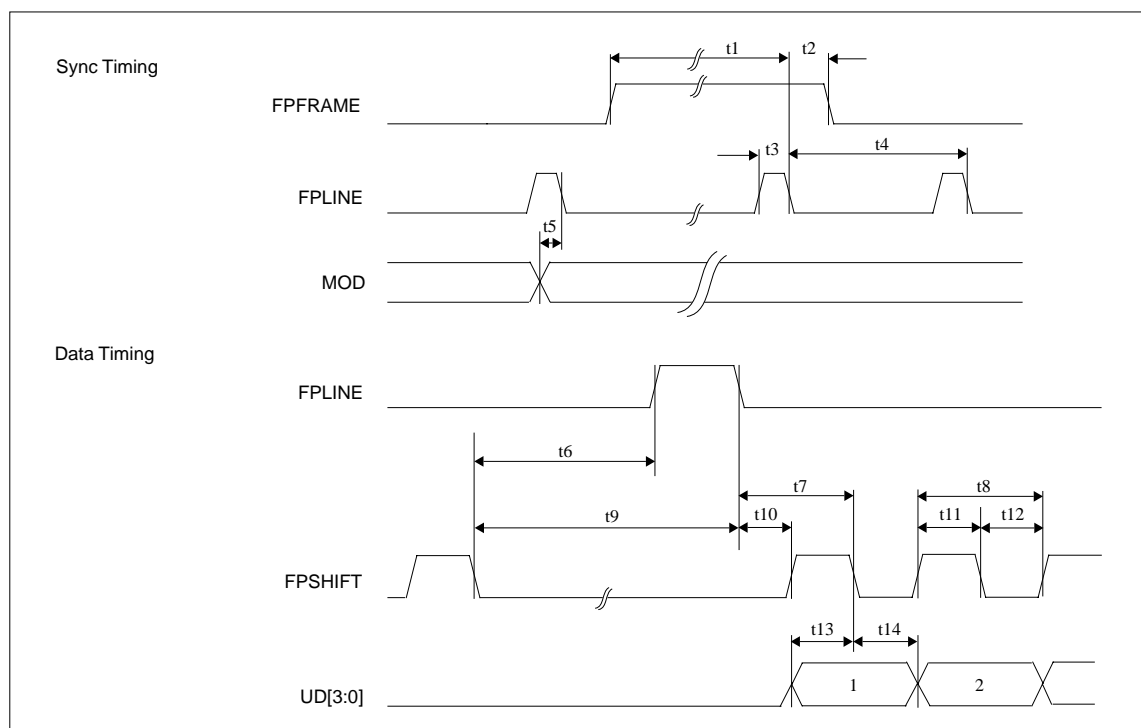


Figure 7-25 4-Bit Single Monochrome Passive LCD Panel A.C. Timing

Table 7-23 4-Bit Single Monochrome Passive LCD Panel A.C. Timing

Symbol	Parameter	Min.	Typ.	Max.	Units
t1	FPFRAME setup to FPLINE pulse trailing edge	note 2			
t2	FPFRAME hold from FPLINE pulse trailing edge	14			Ts (note 1)
t3	FPLINE pulse width	9			Ts
t4	FPLINE period	note 3			
t5	MOD transition to FPLINE pulse trailing edge	1		note 4	
t6	FPSHIFT falling edge to FPLINE pulse leading edge	note 5			
t7	FPLINE pulse trailing edge to FPSHIFT falling edge	t10 + t11			Ts
t8	FPSHIFT period	4			Ts
t9	FPSHIFT falling edge to FPLINE pulse trailing edge	note 6			
t10	FPLINE pulse trailing edge to FPSHIFT rising edge	20			Ts
t11	FPSHIFT pulse width high	2			Ts
t12	FPSHIFT pulse width low	2			Ts
t13	UD[3:0] setup to FPSHIFT falling edge	2			Ts
t14	UD[3:0] hold to FPSHIFT falling edge	2			Ts

Notes: 1. Ts = pixel clock period = memory clock, [memory clock]/2, [memory clock]/3, [memory clock]/4 (see REG[19h] bits [1:0])

2. $t_{1min} = t_{4min} - 14Ts$

3. $t_{4min} = [(((REG[04h] \text{ bits } [6:0]) + 1) * 8 + ((REG[05h] \text{ bits } [4:0]) + 1) * 8) + 33 Ts$

4. $t_{5min} = [(((REG[04h] \text{ bits } [6:0]) + 1) * 8 + ((REG[05h] \text{ bits } [4:0]) + 1) * 8) - 1 Ts$

5. $t_{6min} = [(((REG[05h] \text{ bits } [4:0]) + 1) * 8 - 27) Ts$

6. $t_{9min} = [(((REG[05h] \text{ bits } [4:0]) + 1) * 8 - 18) Ts$

8-Bit Single Monochrome Passive LCD Panel Timing

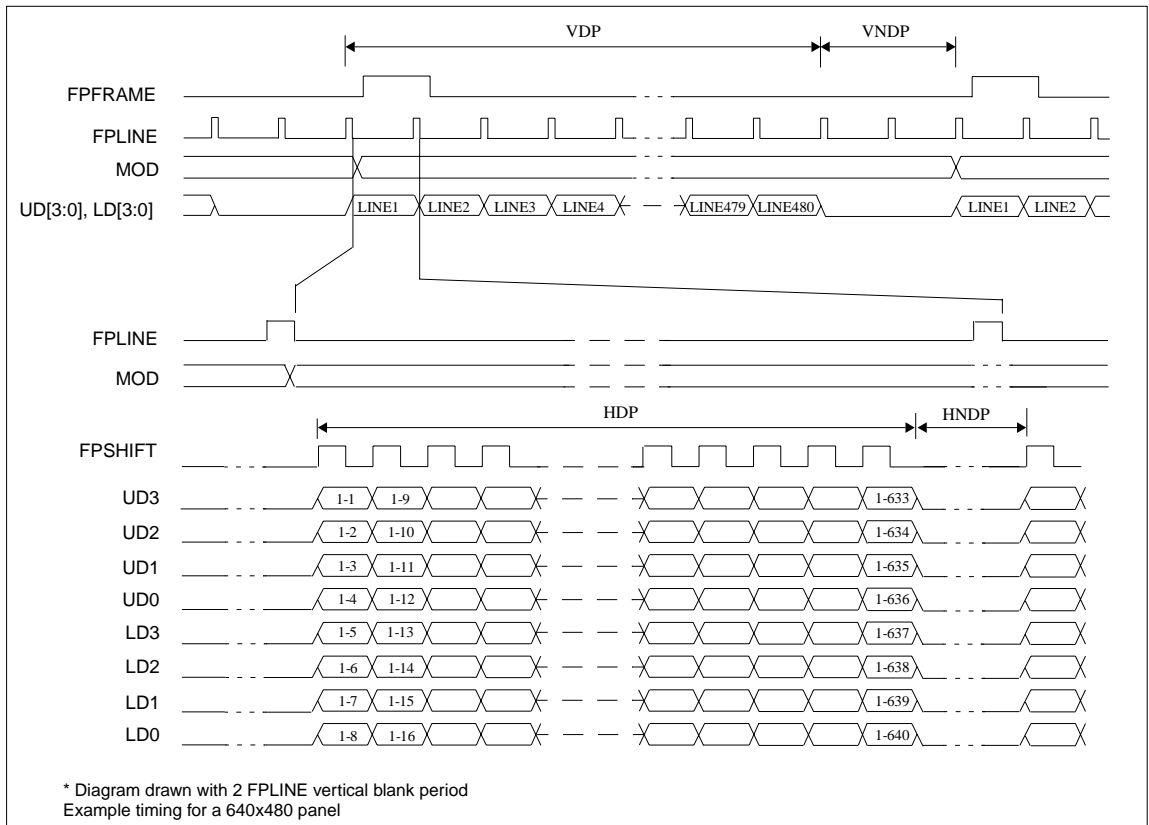


Figure 7-26 8-Bit Single Monochrome Passive LCD Panel Timing

VDP = Vertical Display Period = (REG[09h] bits [1:0], REG[08h] bits [7:0]) + 1
 VNDP = Vertical Non-Display Period = (REG[0Ah] bits [5:0]) + 1
 HDP = Horizontal Display Period = ((REG[04h] bits [6:0]) + 1)*8Ts
 HNDP = Horizontal Non-Display Period = ((REG[05h] bits [4:0]) + 1)*8Ts

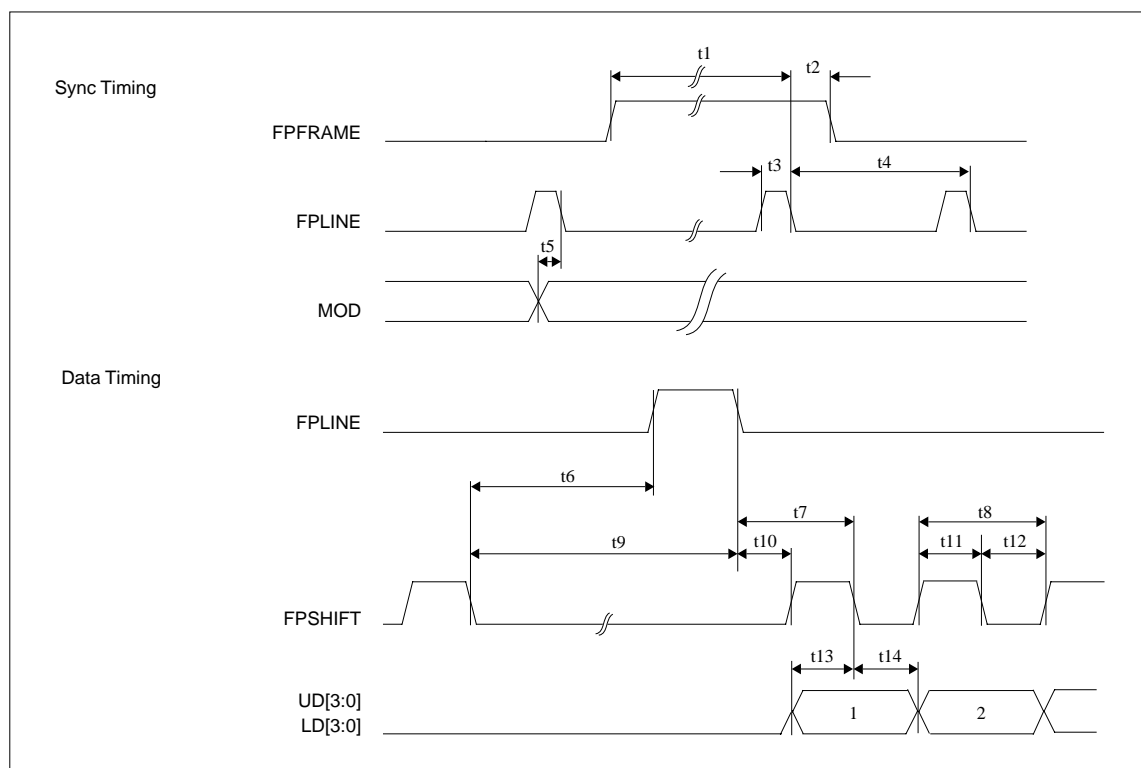


Figure 7-27 8-Bit Single Monochrome Passive LCD Panel A.C. Timing

Table 7-24 8-Bit Single Monochrome Passive LCD Panel A.C. Timing

Symbol	Parameter	Min.	Typ.	Max.	Units
t1	FPFRAME setup to FPLINE pulse trailing edge	note 2			
t2	FPFRAME hold from FPLINE pulse trailing edge	14			Ts (note 1)
t3	FPLINE pulse width	9			Ts
t4	FPLINE period	note 3			
t5	MOD transition to FPLINE pulse trailing edge	1		note 4	
t6	FPSHIFT falling edge to FPLINE pulse leading edge	note 5			
t7	FPLINE pulse trailing edge to FPSHIFT falling edge	t10 + t11			Ts
t8	FPSHIFT period	8			Ts
t9	FPSHIFT falling edge to FPLINE pulse trailing edge	note 6			
t10	FPLINE pulse trailing edge to FPSHIFT rising edge	20			Ts
t11	FPSHIFT pulse width high	4			Ts
t12	FPSHIFT pulse width low	4			Ts
t13	UD[3:0], LD[3:0] setup to FPSHIFT falling edge	4			Ts
t14	UD[3:0], LD[3:0] hold to FPSHIFT falling edge	4			Ts

Notes: 1. Ts = pixel clock period = memory clock, [memory clock]/2, [memory clock]/3, [memory clock]/4 (see REG[19h] bits [1:0])

2. $t_{1min} = t_{4min} - 14Ts$

3. $t_{4min} = [(((REG[04h] \text{ bits } [6:0]) + 1) * 8 + ((REG[05h] \text{ bits } [4:0]) + 1) * 8) + 33] Ts$

4. $t_{5min} = [(((REG[04h] \text{ bits } [6:0]) + 1) * 8 + ((REG[05h] \text{ bits } [4:0]) + 1) * 8) - 1] Ts$

5. $t_{6min} = [(((REG[05h] \text{ bits } [4:0]) + 1) * 8 - 25)] Ts$

6. $t_{9min} = [(((REG[05h] \text{ bits } [4:0]) + 1) * 8 - 16)] Ts$

4-Bit Single Color Passive LCD Panel Timing

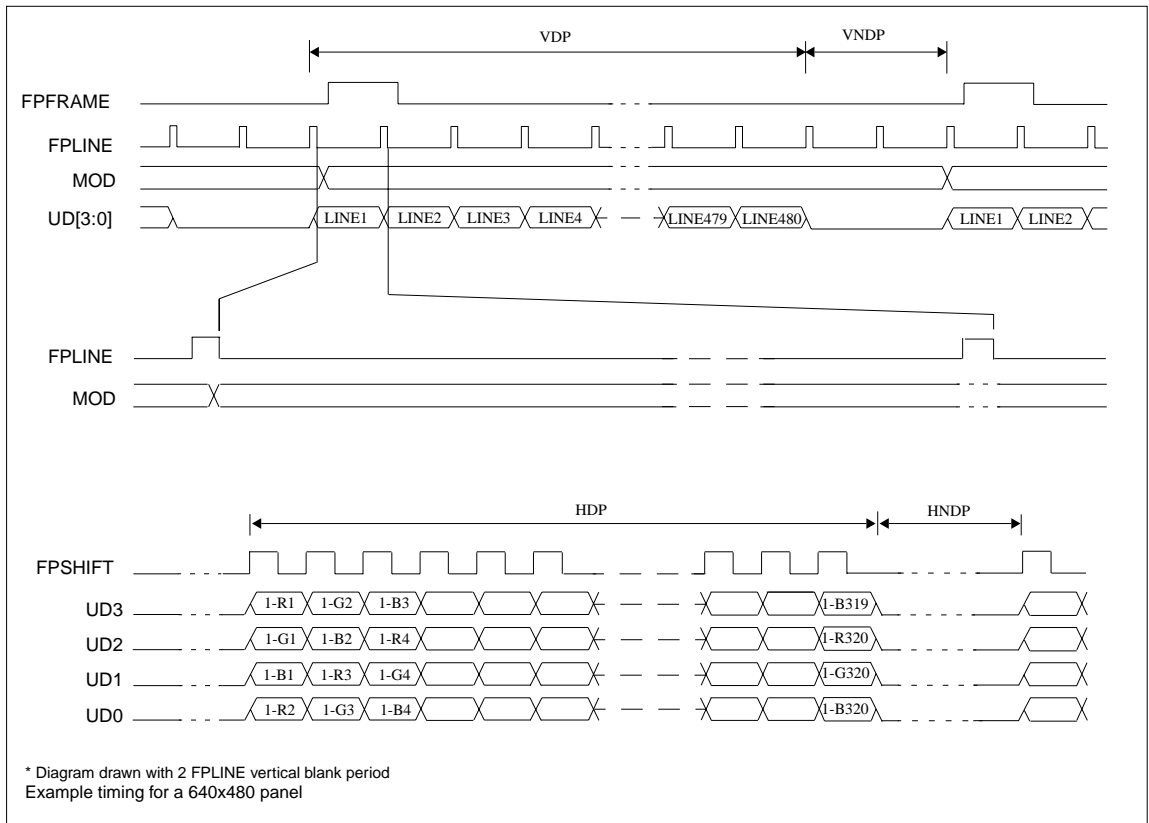


Figure 7-28 4-Bit Single Color Passive LCD Panel Timing

$VDP = \text{Vertical Display Period} = (\text{REG}[09\text{h}] \text{ bits } [1:0], \text{REG}[08\text{h}] \text{ bits } [7:0]) + 1$
 $VNDP = \text{Vertical Non-Display Period} = (\text{REG}[0A\text{h}] \text{ bits } [5:0]) + 1$
 $HDP = \text{Horizontal Display Period} = ((\text{REG}[04\text{h}] \text{ bits } [6:0]) + 1) * 8T_s$
 $HNDP = \text{Horizontal Non-Display Period} = ((\text{REG}[05\text{h}] \text{ bits } [4:0]) + 1) * 8T_s$

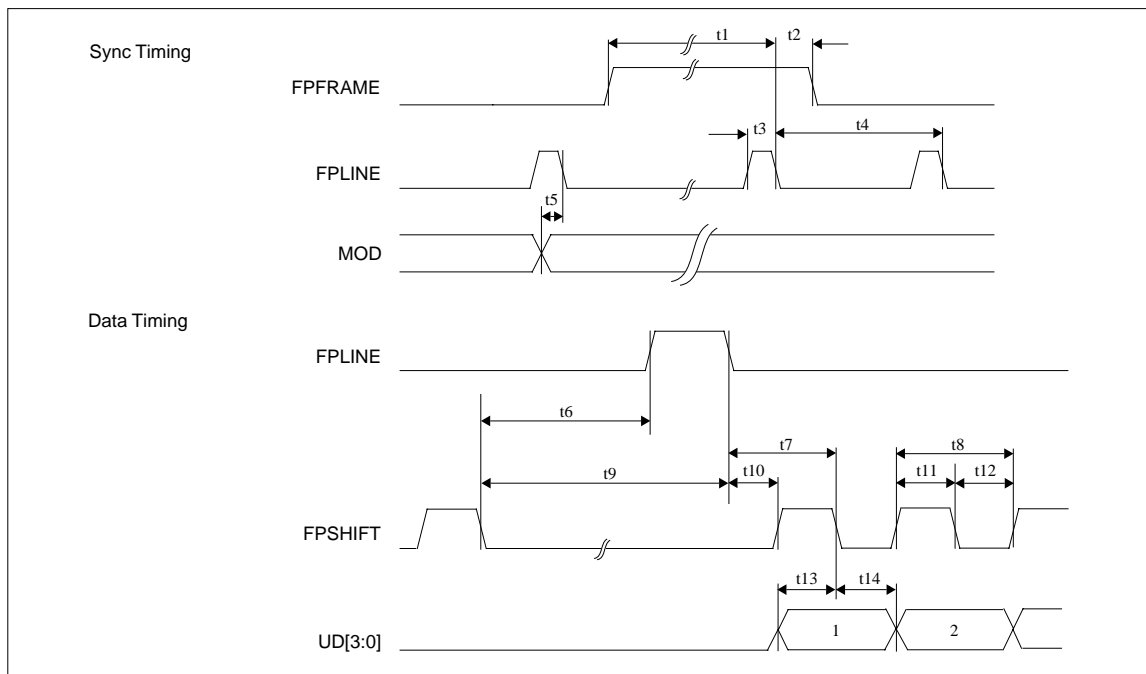


Figure 7-29 4-Bit Single Color Passive LCD Panel A.C.Timing

Table 7-25 4-Bit Single Color Passive LCD Panel A.C.Timing

Symbol	Parameter	Min.	Typ.	Max.	Units
t1	FPFRAME setup to FPLINE pulse trailing edge	note 2			
t2	FPFRAME hold from FPLINE pulse trailing edge	14			Ts (note 1)
t3	FPLINE pulse width	9			Ts
t4	FPLINE period	note 3			
t5	MOD transition to FPLINE pulse trailing edge	1		note 4	
t6	FPSHIFT falling edge to FPLINE pulse leading edge	note 5			
t7	FPLINE pulse trailing edge to FPSHIFT falling edge	t10 + t11			Ts
t8	FPSHIFT period	4			Ts
t9	FPSHIFT falling edge to FPLINE pulse trailing edge	note 6			
t10	FPLINE pulse trailing edge to FPSHIFT rising edge	21			Ts
t11	FPSHIFT pulse width high	2			Ts
t12	FPSHIFT pulse width low	2			Ts
t13	UD[3:0], setup to FPSHIFT falling edge	2			Ts
t14	UD[3:0], hold from FPSHIFT falling edge	2			Ts

Notes: 1. Ts = pixel clock period = memory clock, [memory clock]/2, [memory clock]/3, [memory clock]/4 (see REG[19h] bits [1:0])

2. $t_{1min} = t_{4min} - 14Ts$

3. $t_{4min} = [((REG[04h] \text{ bits } [6:0]) + 1) * 8 + ((REG[05h] \text{ bits } [4:0]) + 1) * 8] + 33 Ts$

4. $t_{5min} = [(((REG[04h] \text{ bits } [6:0]) + 1) * 8 + ((REG[05h] \text{ bits } [4:0]) + 1) * 8) - 1] Ts$

5. $t_{6min} = [((REG[05h] \text{ bits } [4:0]) + 1) * 8 - 28] Ts$

6. $t_{9min} = [((REG[05h] \text{ bits } [4:0]) + 1) * 8 - 19] Ts$

8-Bit Single Color Passive LCD Panel Timing (Format 1)

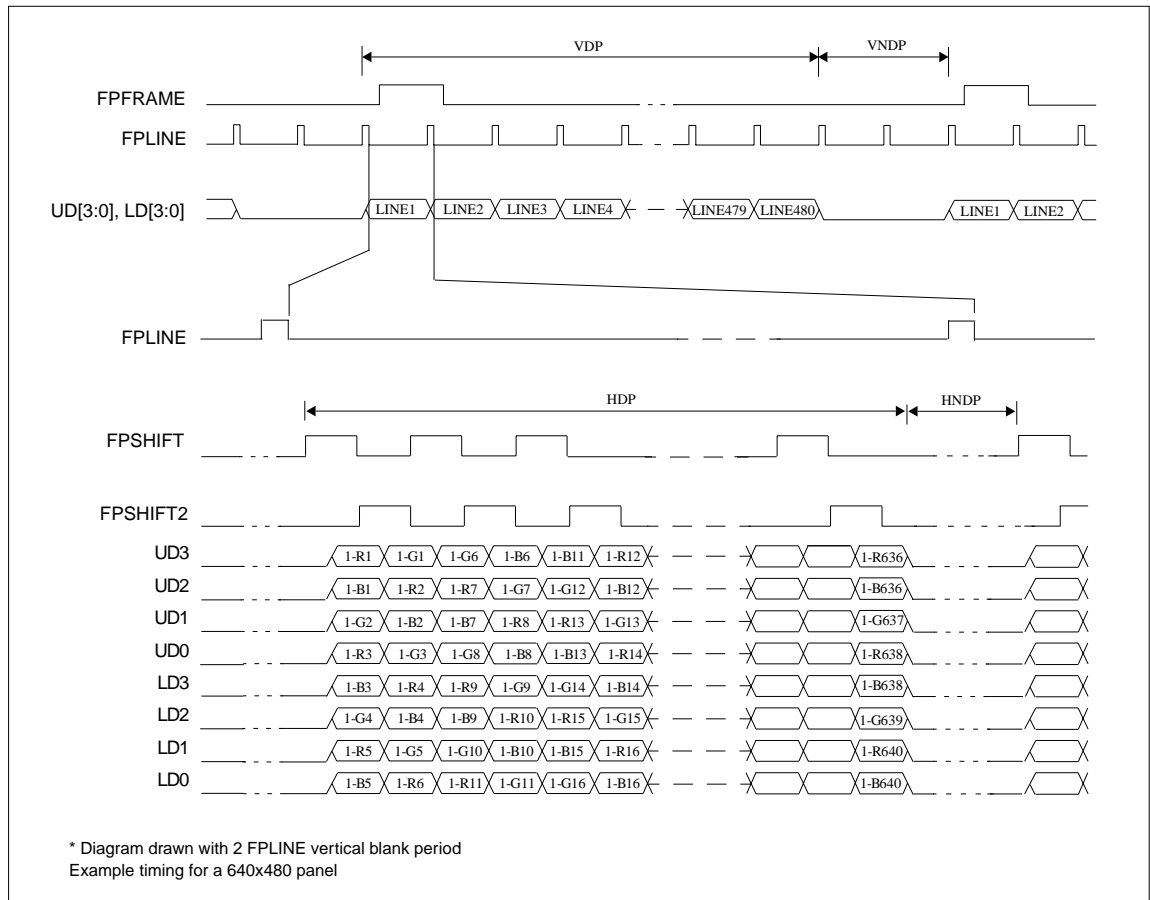


Figure 7-30 8-Bit Single Color Passive LCD Panel Timing (Format 1)

$VDP = \text{Vertical Display Period} = (\text{REG}[09h] \text{ bits } [1:0], \text{REG}[08h] \text{ bits } [7:0]) + 1$
 $VNDP = \text{Vertical Non-Display Period} = (\text{REG}[0Ah] \text{ bits } [5:0]) + 1$
 $HDP = \text{Horizontal Display Period} = ((\text{REG}[04h] \text{ bits } [6:0]) + 1) * 8Ts$
 $HNDP = \text{Horizontal Non-Display Period} = ((\text{REG}[05h] \text{ bits } [4:0]) + 1) * 8Ts$

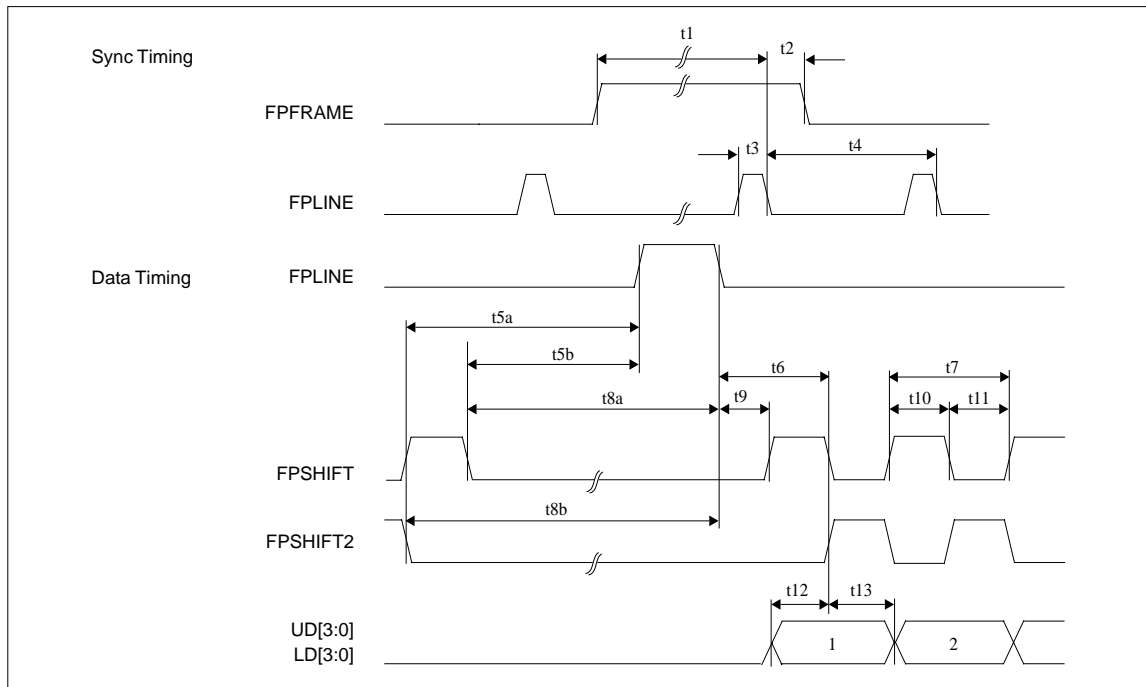


Figure 7-31 8-Bit Single Color Passive LCD Panel A.C. Timing (Format 1)

Table 7-26 8-Bit Single Color Passive LCD Panel A.C. Timing (Format 1)

Symbol	Parameter	Min.	Typ.	Max.	Units
t1	FPPFRAME setup to FPLINE pulse trailing edge	note 2			
t2	FPPFRAME hold from FPLINE pulse trailing edge	14			Ts (note 1)
t3	FPLINE pulse width	9			Ts
t4	FPLINE period	note 3			
t5a	FPSHIFT2 falling edge to FPLINE pulse leading edge	note 4			
t5b	FPSHIFT falling edge to FPLINE pulse leading edge	note 5			
t6	FPLINE pulse trailing edge to FPSHIFT2 rising, FPSHIFT falling edge	t9 + t10			Ts
t7	FPSHIFT2, FPSHIFT period	4			Ts
t8a	FPSHIFT2 falling edge to FPLINE pulse trailing edge	note 6			
t8b	FPSHIFT2 falling edge to FPLINE pulse trailing edge	note 7			
t9	FPLINE pulse trailing edge to FPSHIFT rising edge	20			Ts
t10	FPSHIFT2, FPSHIFT pulse width high	2			Ts
t11	FPSHIFT2, FPSHIFT pulse width low	2			Ts
t12	UD[3:0], LD[3:0] setup to FPSHIFT2 rising, FPSHIFT falling edge	1			Ts
t13	UD[3:0], LD[3:0] hold from FPSHIFT2 rising, FPSHIFT falling edge	1			Ts

Notes: 1. Ts = pixel clock period = memory clock, [memory clock]/2, [memory clock]/3, [memory clock]/4 (see REG[19h] bits [1:0])

2. t1min = t4min - 14Ts

3. t4min = [((REG[04h] bits [6:0]) + 1) * 8 + ((REG[05h] bits [4:0]) + 1) * 8] Ts

4. t5min = [((REG[05h] bits [4:0]) + 1) * 8 - 27] Ts

5. t5min = [((REG[05h] bits [4:0]) + 1) * 8 - 29] Ts

6. t8min = [((REG[05h] bits [4:0]) + 1) * 8 - 20] Ts

7. t8min = [((REG[05h] bits [4:0]) + 1) * 8 - 18] Ts

8-Bit Single Color Passive LCD Panel Timing (Format 2)

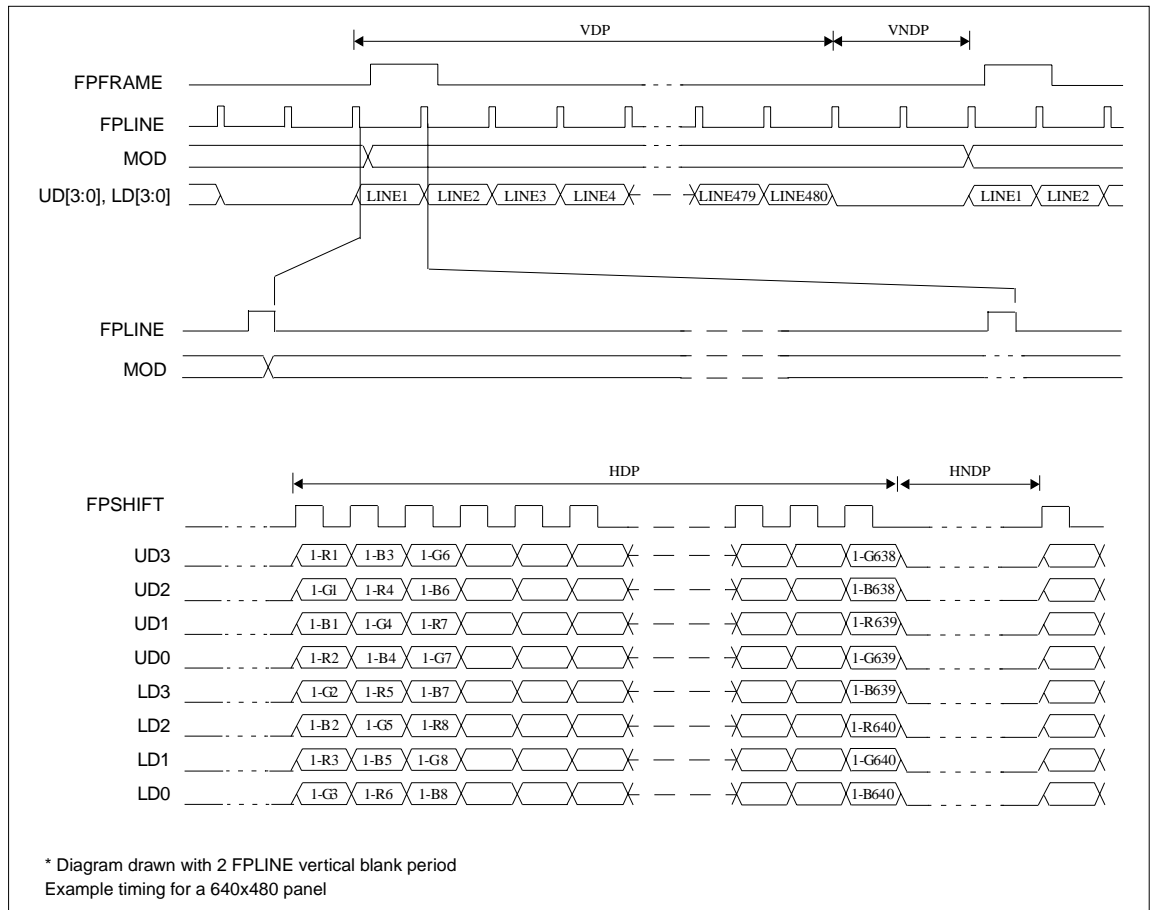


Figure 7-32 8-Bit Single Color Passive LCD Panel Timing (Format 2)

$VDP = \text{Vertical Display Period} = (\text{REG}[09\text{h}] \text{ bits } [1:0], \text{REG}[08\text{h}] \text{ bits } [7:0]) + 1$
 $VNDP = \text{Vertical Non-Display Period} = (\text{REG}[0A\text{h}] \text{ bits } [5:0]) + 1$
 $HDP = \text{Horizontal Display Period} = ((\text{REG}[04\text{h}] \text{ bits } [6:0]) + 1) * 8Ts$
 $HNDP = \text{Horizontal Non-Display Period} = ((\text{REG}[05\text{h}] \text{ bits } [4:0]) + 1) * 8Ts$

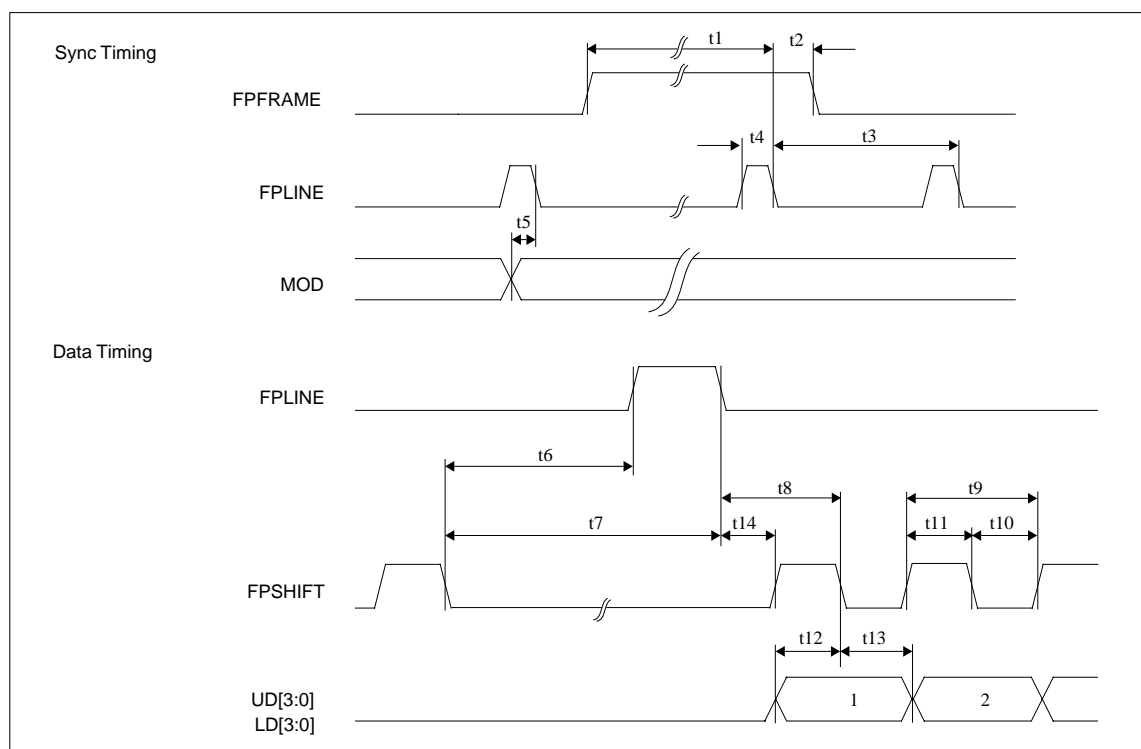


Figure 7-33 8-Bit Single Color Passive LCD Panel A.C. Timing (Format 2)

Table 7-27 8-Bit Single Color Passive LCD Panel A.C. Timing (Format 2)

Symbol	Parameter	Min.	Typ.	Max.	Units
t1	FPFRAME setup to FPLINE pulse trailing edge	note 2			
t2	FPFRAME hold from FPLINE pulse trailing edge	14			Ts (note 1)
t3	FPLINE period	note 3			
t4	FPLINE pulse width	9			Ts
t5	MOD transition to FPLINE pulse trailing edge	1		note 4	
t6	FPSHIFT falling edge to FPLINE pulse leading edge	note 5			
t7	FPSHIFT falling edge to FPLINE pulse trailing edge	note 6			
t8	FPLINE pulse trailing edge to FPSHIFT falling edge	t14 + 2			
t9	FPSHIFT period	2			Ts
t10	FPSHIFT pulse width low	1			Ts
t11	FPSHIFT pulse width high	1			Ts
t12	UD[3:0], LD[3:0] setup to FPSHIFT falling edge	1			Ts
t13	UD[3:0], LD[3:0] hold to FPSHIFT falling edge	1			Ts
t14	FPLINE pulse trailing edge to FPSHIFT rising edge	20			Ts

Notes: 1. Ts = pixel clock period = memory clock, [memory clock]/2, [memory clock]/3, [memory clock]/4 (see REG[19h] bits [1:0])

2. $t1_{min} = t3_{min} - 14Ts$

3. $t3_{min} = [(((REG[04h] \text{ bits } [6:0]) + 1) * 8 + ((REG[05h] \text{ bits } [4:0]) + 1) * 8) + 33] Ts$

4. $t5_{min} = [(((REG[04h] \text{ bits } [6:0]) + 1) * 8 + ((REG[05h] \text{ bits } [4:0]) + 1) * 8) - 1] Ts$

5. $t6_{min} = [(((REG[05h] \text{ bits } [4:0]) + 1) * 8 - 28)] Ts$

6. $t7_{min} = [(((REG[05h] \text{ bits } [4:0]) + 1) * 8 - 19)] Ts$

16-Bit Single Color Passive LCD Panel Timing

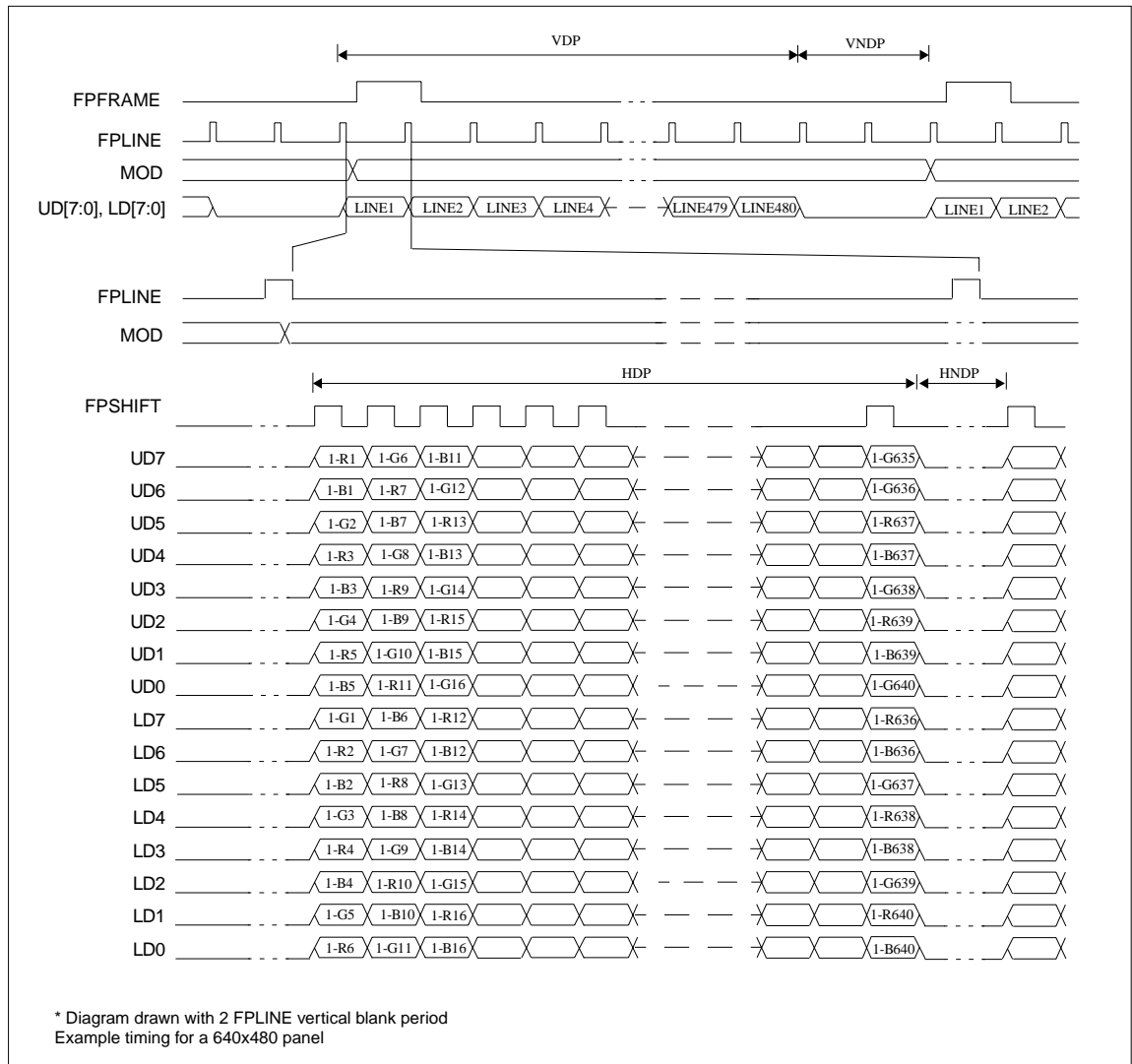


Figure 7-34 16-Bit Single Color Passive LCD Panel Timing

VDP = Vertical Display Period = (REG[09h] bits [1:0], REG[08h] bits [7:0]) + 1
 VNDP = Vertical Non-Display Period = (REG[0Ah] bits [5:0]) + 1
 HDP = Horizontal Display Period = ((REG[04h] bits [6:0]) + 1)*8Ts
 HNDP = Horizontal Non-Display Period = ((REG[05h] bits [4:0]) + 1)*8Ts

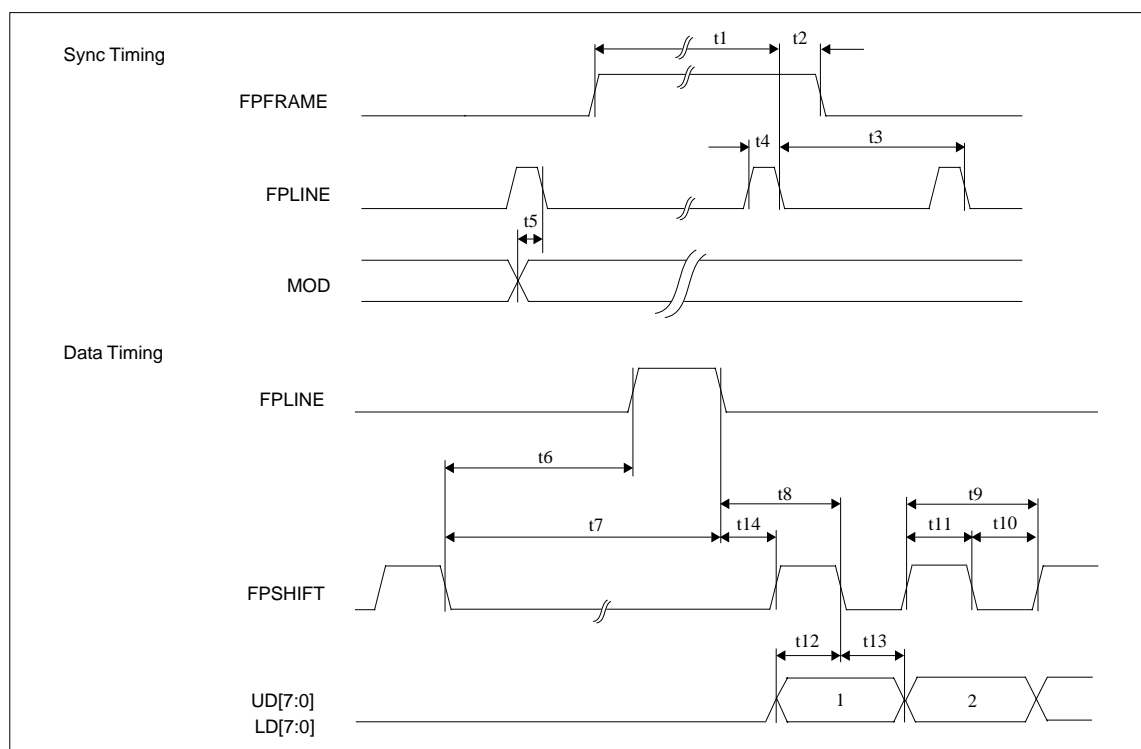


Figure 7-35 16-Bit Single Color Passive LCD Panel A.C. Timing

Table 7-28 16-Bit Single Color Passive LCD Panel A.C. Timing

Symbol	Parameter	Min.	Typ.	Max.	Units
t1	FPFRAME setup to FPLINE pulse trailing edge	note 2			
t2	FPFRAME hold from FPLINE pulse trailing edge	14			Ts (note 1)
t3	FPLINE period	note 3			
t4	FPLINE pulse width	9			Ts
t5	MOD transition to FPLINE pulse trailing edge	1		note 4	
t6	FPSHIFT falling edge to FPLINE pulse leading edge	note 5			
t7	FPSHIFT falling edge to FPLINE pulse trailing edge	note 6			
t8	FPLINE pulse trailing edge to FPSHIFT falling edge	$t_{14} + 3$			Ts
t9	FPSHIFT period	5			Ts
t10	FPSHIFT pulse width low	2			Ts
t11	FPSHIFT pulse width high	2			Ts
t12	UD[7:0], LD[7:0] setup to FPSHIFT falling edge	2			Ts
t13	UD[7:0], LD[7:0] hold to FPSHIFT falling edge	2			Ts
t14	FPLINE pulse trailing edge to FPSHIFT rising edge	20			Ts

Notes: 1. Ts = pixel clock period = memory clock, [memory clock]/2, [memory clock]/3, [memory clock]/4 (see REG[19h] bits [1:0])

2. $t_{1\min} = t_{3\min} - 14Ts$

3. $t_{3\min} = [((REG[04h] \text{ bits } [6:0]) + 1) * 8 + ((REG[05h] \text{ bits } [4:0]) + 1) * 8] + 33 Ts$

4. $t_{5\min} = [(((REG[04h] \text{ bits } [6:0]) + 1) * 8 + ((REG[05h] \text{ bits } [4:0]) + 1) * 8) - 1] Ts$

5. $t_{6\min} = [(REG[05h] \text{ bits } [4:0]) + 1] * 8 - 27 Ts$

6. $t_{7\min} = [((REG[05h] \text{ bits } [4:0]) + 1) * 8 - 18] Ts$

8-Bit Dual Monochrome Passive LCD Panel Timing

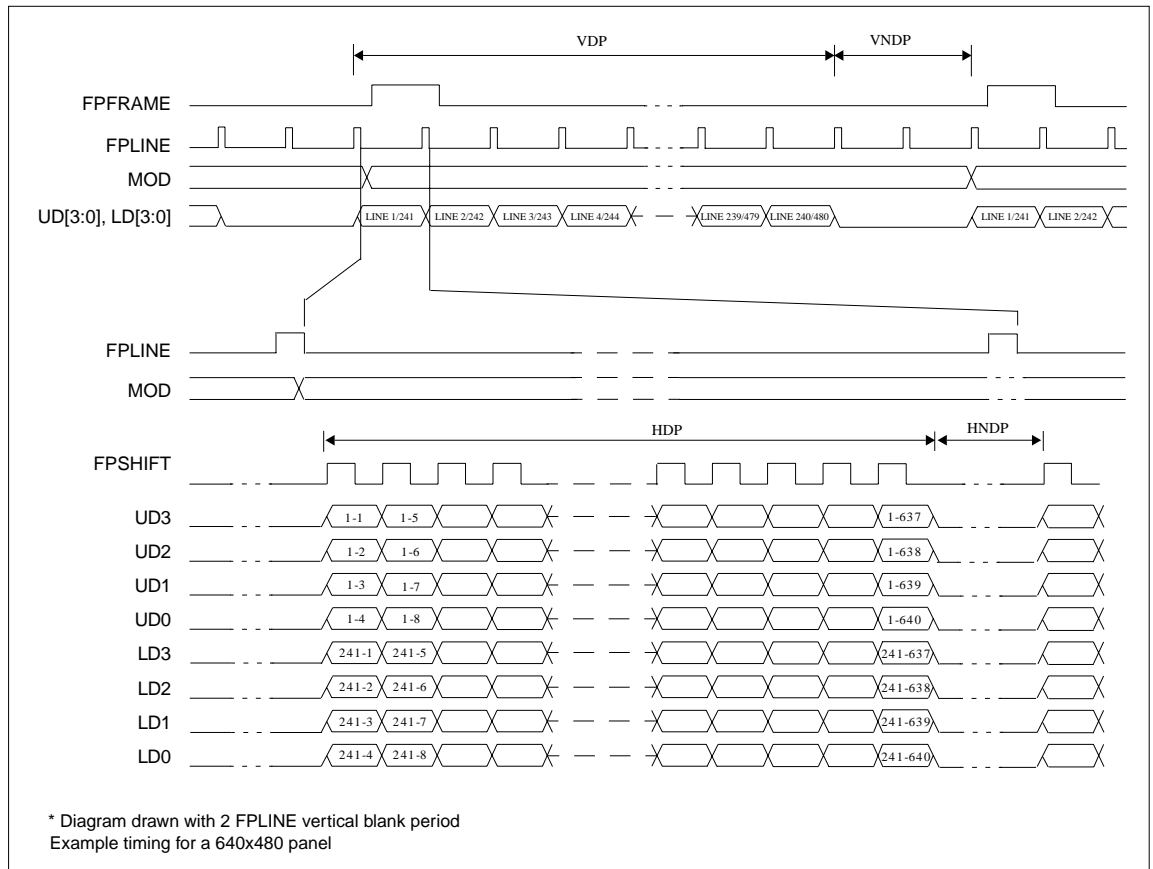


Figure 7-36 8-Bit Dual Monochrome Passive LCD Panel Timing

VDP = Vertical Display Period = (REG[09h] bits [1:0], REG[08h] bits [7:0]) + 1
 VNDP = Vertical Non-Display Period = (REG[0Ah] bits [5:0]) + 1
 HDP = Horizontal Display Period = ((REG[04h] bits [6:0]) + 1)*8Ts
 HNDP = Horizontal Non-Display Period = ((REG[05h] bits [4:0]) + 1)*8Ts

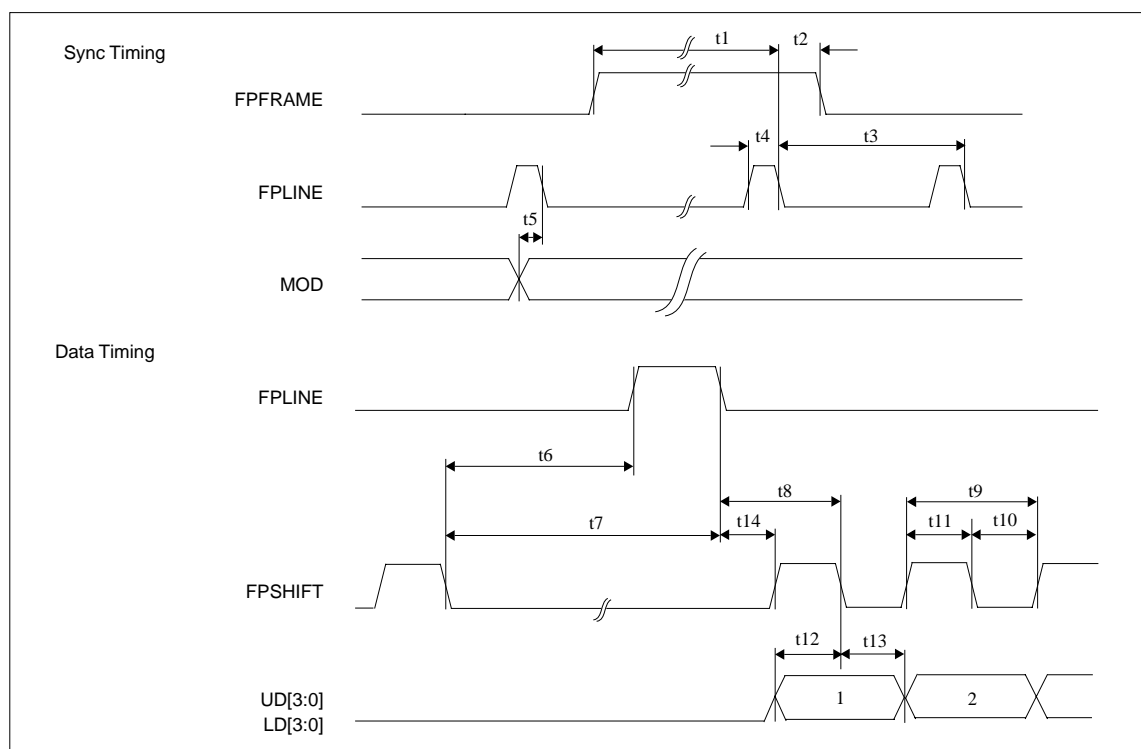


Figure 7-37 8-Bit Dual Monochrome Passive LCD Panel A.C. Timing

Table 7-29 8-Bit Dual Monochrome Passive LCD Panel A.C. Timing

Symbol	Parameter	Min.	Typ.	Max.	Units
t1	FPFRAME setup to FPLINE pulse trailing edge	note 2			
t2	FPFRAME hold from FPLINE pulse trailing edge	14			Ts (note 1)
t3	FPLINE period	note 3			
t4	FPLINE pulse width	9			Ts
t5	MOD transition to FPLINE pulse trailing edge	1		note 4	
t6	FPSHIFT falling edge to FPLINE pulse leading edge	note 5			
t7	FPSHIFT falling edge to FPLINE pulse trailing edge	note 6			
t8	FPLINE pulse trailing edge to FPSHIFT falling edge	t14 + 2			Ts
t9	FPSHIFT period	4			Ts
t10	FPSHIFT pulse width low	2			Ts
t11	FPSHIFT pulse width high	2			Ts
t12	UD[3:0], LD[3:0] setup to FPSHIFT falling edge	2			Ts
t13	UD[3:0], LD[3:0] hold to FPSHIFT falling edge	2			Ts
t14	FPLINE pulse trailing edge to FPSHIFT rising edge	12			Ts

Notes: 1. Ts = pixel clock period = memory clock, [memory clock]/2, [memory clock]/3, [memory clock]/4 (see REG[19h] bits [1:0])

2. $t_{1min} = t_{3min} - 14Ts$

3. $t_{3min} = [(((REG[04h] \text{ bits } [6:0]) + 1) * 8 + ((REG[05h] \text{ bits } [4:0]) + 1) * 8) + 33] Ts$

4. $t_{5min} = [(((REG[04h] \text{ bits } [6:0]) + 1) * 8 + ((REG[05h] \text{ bits } [4:0]) + 1) * 8) - 1] Ts$

5. $t_{6min} = [(((REG[05h] \text{ bits } [4:0]) + 1) * 8 - 19)] Ts$

6. $t_{7min} = [(((REG[05h] \text{ bits } [4:0]) + 1) * 8 - 10)] Ts$

1000



1000

1000

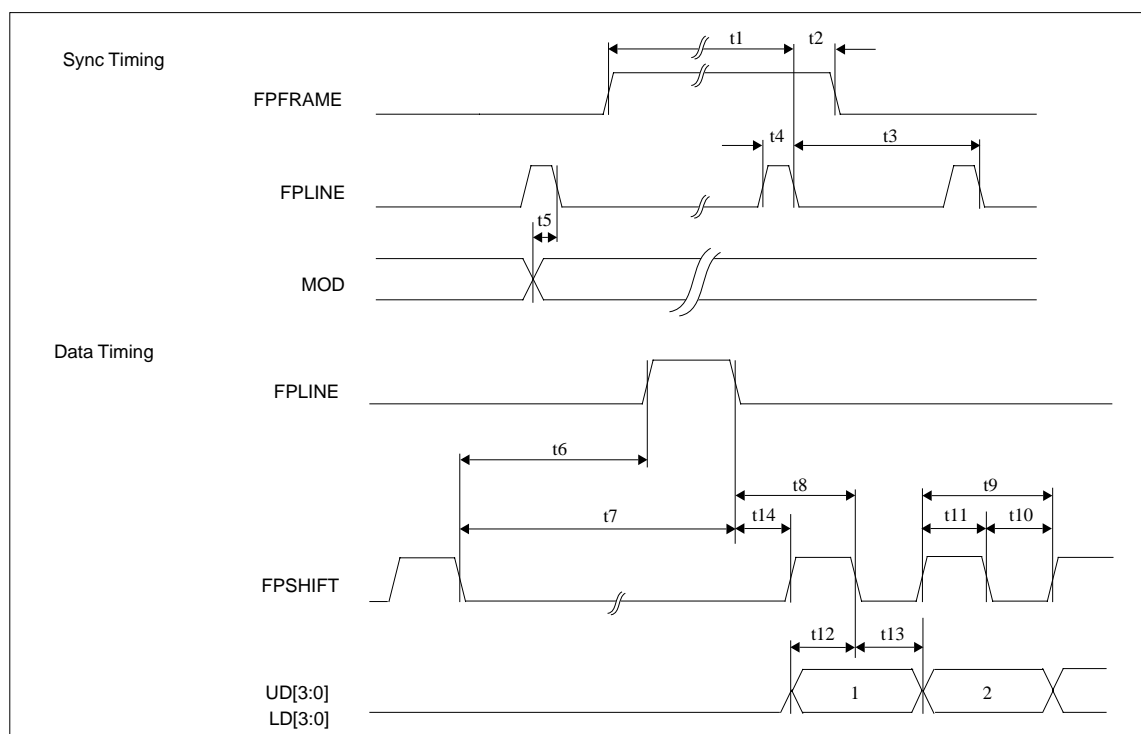


Figure 7-39 8-Bit Dual Color Passive LCD Panel A.C. Timing

Table 7-30 8-Bit Dual Color Passive LCD Panel A.C. Timing

Symbol	Parameter	Min.	Typ.	Max.	Units
t1	FFRAME setup to FPLINE pulse trailing edge	note 2			
t2	FFRAME hold from FPLINE pulse trailing edge	14			Ts (note 1)
t3	FPLINE period	note 3			
t4	FPLINE pulse width	9			Ts
t5	MOD transition to FPLINE pulse trailing edge	1		note 4	
t6	FPSHIFT falling edge to FPLINE pulse leading edge	note 5			
t7	FPSHIFT falling edge to FPLINE pulse trailing edge	note 6			
t8	FPLINE pulse trailing edge to FPSHIFT falling edge	t14 + t11			Ts
t9	FPSHIFT period	1			Ts
t10	FPSHIFT pulse width low	0.45			Ts
t11	FPSHIFT pulse width high	0.45			Ts
t12	UD[3:0], LD[3:0] setup to FPSHIFT falling edge	0.45			Ts
t13	UD[3:0], LD[3:0] hold to FPSHIFT falling edge	0.45			Ts
t14	FPLINE pulse trailing edge to FPSHIFT rising edge	13			Ts

Notes: 1. Ts = pixel clock period = memory clock, [memory clock]/2, [memory clock]/3, [memory clock]/4 (see REG[19h] bits [1:0])

2. $t1_{min} = t3_{min} - 14Ts$

3. $t3_{min} = [(((REG[04h] \text{ bits } [6:0]) + 1) * 8 + ((REG[05h] \text{ bits } [4:0]) + 1) * 8) + 33] Ts$

4. $t5_{min} = [(((REG[04h] \text{ bits } [6:0]) + 1) * 8 + ((REG[05h] \text{ bits } [4:0]) + 1) * 8) - 1] Ts$

5. $t6_{min} = [((REG[05h] \text{ bits } [4:0]) + 1) * 8 - 20] Ts$

6. $t7_{min} = [((REG[05h] \text{ bits } [4:0]) + 1) * 8 - 11] Ts$

16-Bit Dual Color Passive LCD Panel Timing

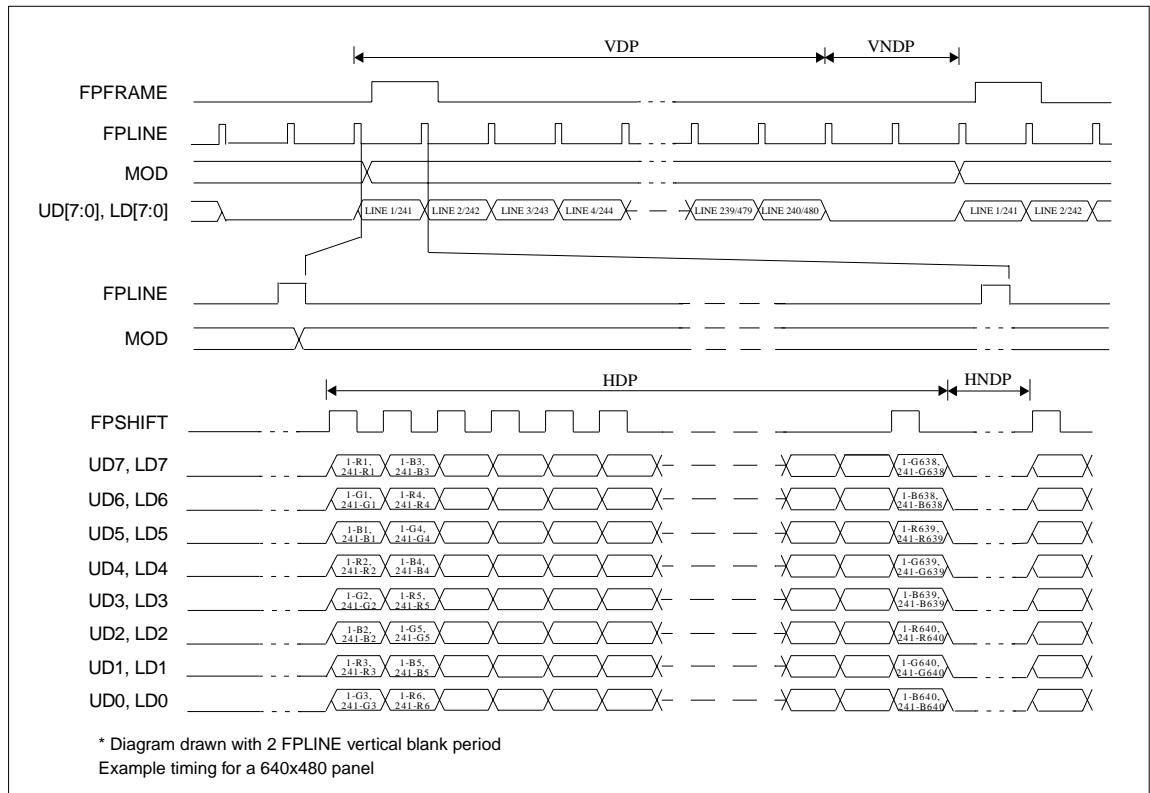


Figure 7-40 16-Bit Dual Color Passive LCD Panel Timing

VDP = Vertical Display Period = (REG[09h] bits [1:0], REG[08h] bits [7:0]) + 1
 VNDP = Vertical Non-Display Period = (REG[0Ah] bits [5:0]) + 1
 HDP = Horizontal Display Period = ((REG[04h] bits [6:0]) + 1)*8Ts
 HN = Horizontal Non-Display Period = ((REG[05h] bits [4:0]) + 1)*8Ts

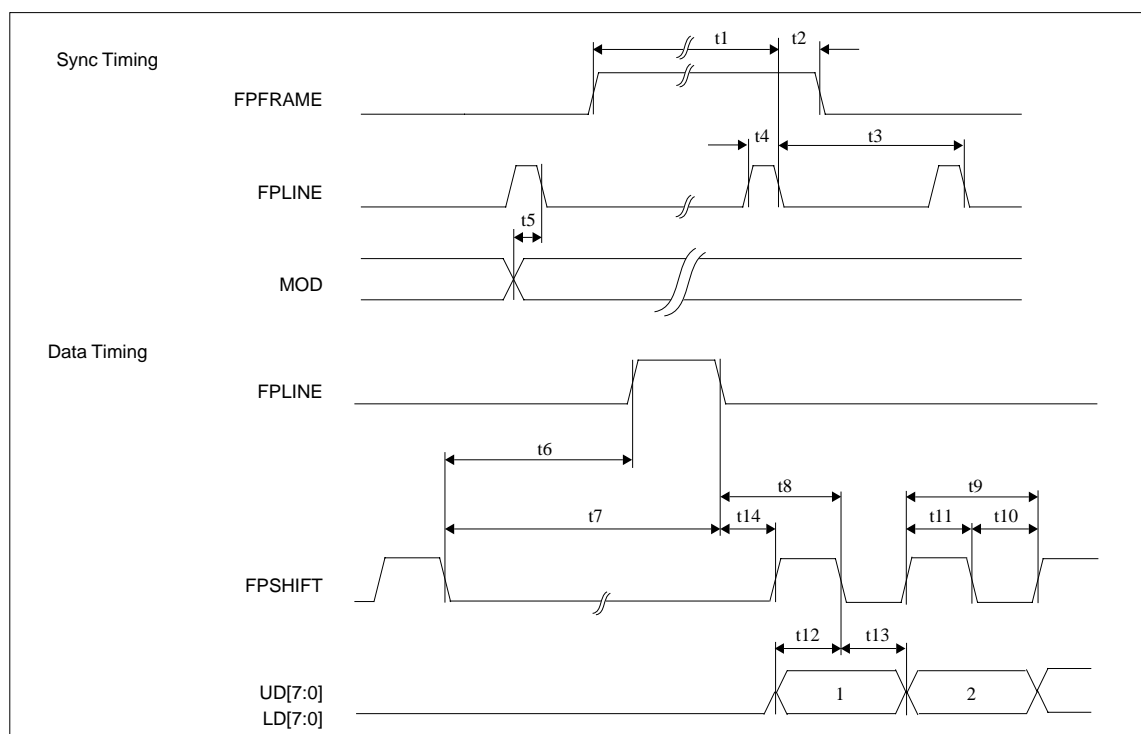


Figure 7-41 16-Bit Dual Color Passive LCD Panel A.C. Timing

Table 7-31 16-Bit Dual Color Passive LCD Panel A.C. Timing

Symbol	Parameter	Min.	Typ.	Max.	Units
t1	FPFRAME setup to FPLINE pulse trailing edge	note 2			
t2	FPFRAME hold from FPLINE pulse trailing edge	14			Ts (note 1)
t3	FPLINE period	note 3			
t4	FPLINE pulse width	9			Ts
t5	MOD transition to FPLINE pulse trailing edge	1		note 4	
t6	FPSHIFT falling edge to FPLINE pulse leading edge	note 5			
t7	FPSHIFT falling edge to FPLINE pulse trailing edge	note 6			
t8	FPLINE pulse trailing edge to FPSHIFT falling edge	t14 + 2			
t9	FPSHIFT period	2			Ts
t10	FPSHIFT pulse width low	1			Ts
t11	FPSHIFT pulse width high	1			Ts
t12	UD[7:0], LD[7:0] setup to FPSHIFT falling edge	1			Ts
t13	UD[7:0], LD[7:0] hold to FPSHIFT falling edge	1			Ts
t14	FPLINE pulse trailing edge to FPSHIFT rising edge	12			Ts

Notes: 1. Ts = pixel clock period = memory clock, [memory clock]/2, [memory clock]/3, [memory clock]/4 (see REG[19h] bits [1:0])

2. $t1_{min} = t3_{min} - 14Ts$

3. $t3_{min} = [(((REG[04h] \text{ bits } [6:0]) + 1) * 8 + ((REG[05h] \text{ bits } [4:0]) + 1) * 8) + 33] Ts$

4. $t5_{min} = [(((REG[04h] \text{ bits } [6:0]) + 1) * 8 + ((REG[05h] \text{ bits } [4:0]) + 1) * 8) - 1] Ts$

5. $t6_{min} = [((REG[05h] \text{ bits } [4:0]) + 1) * 8 - 20] Ts$

6. $t7_{min} = [((REG[05h] \text{ bits } [4:0]) + 1) * 8 - 11] Ts$

16-Bit TFT/D-TFD Panel Timing

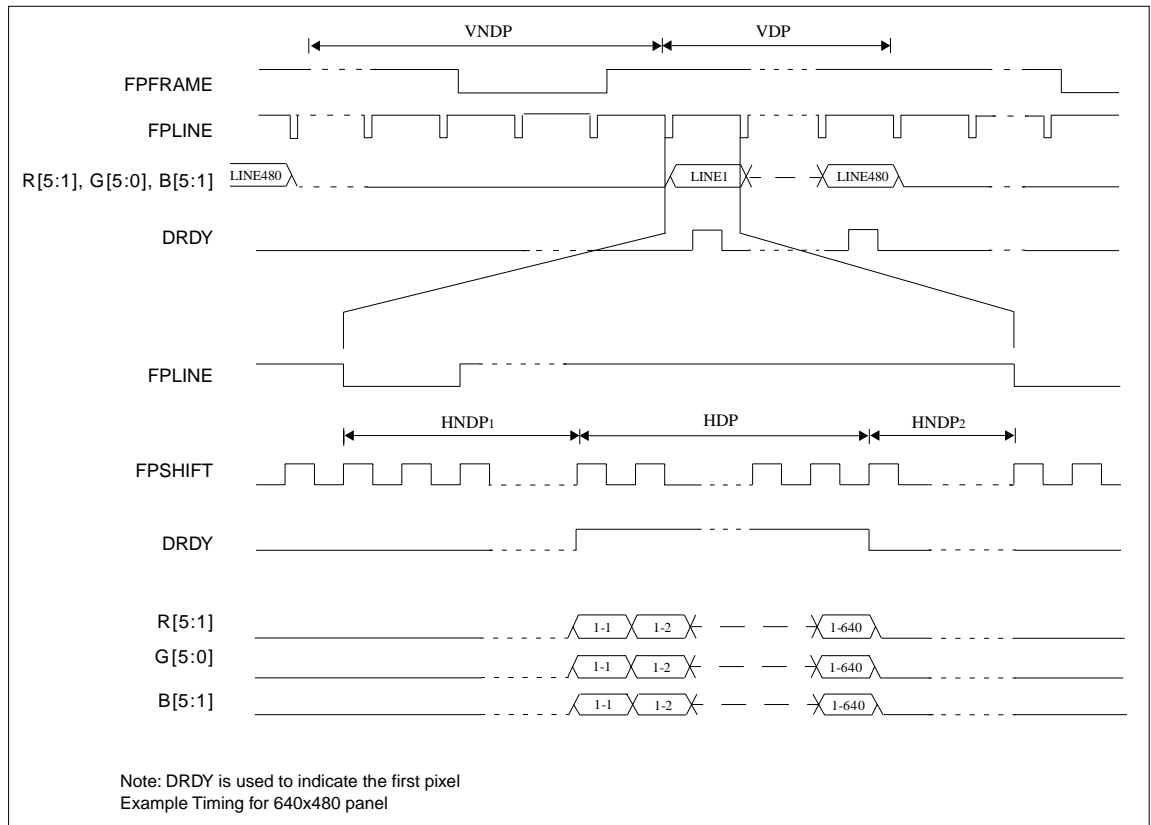


Figure 7-42 16-Bit TFT/D-TFD Panel Timing

VDP	= Vertical Display Period	= (REG[09h] bits [1:0], REG[08h] bits [7:0]) + 1
VNDP	= Vertical Non-Display Period	= (REG[0Ah] bits [5:0]) + 1
HDP	= Horizontal Display Period	= ((REG[04h] bits [6:0]) + 1)*8Ts
HNDP	= Horizontal Non-Display Period	= HNDP ₁ + HNDP ₂ = ((REG[05h] bits [4:0]) + 1)*8Ts

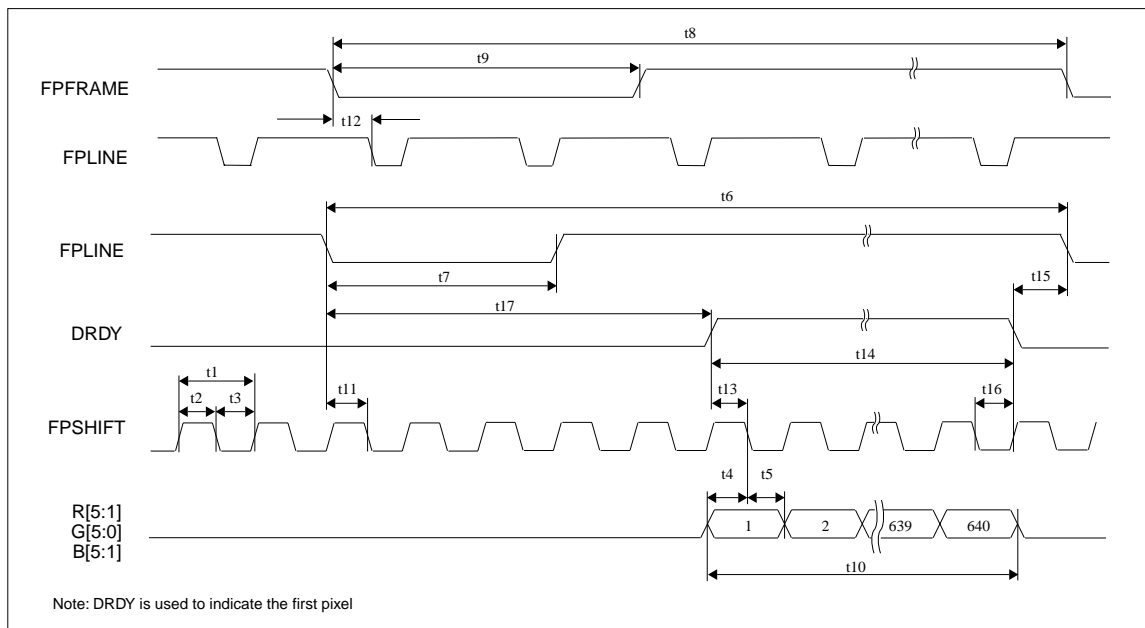


Figure 7-43 16-Bit TFT/D-TFD A.C. Timing

Table 7-32 16-Bit TFT/D-TFD A.C. Timing

Symbol	Parameter	Min.	Typ.	Max.	Units
t1	FPSHIFT period	1			Ts (note 1)
t2	FPSHIFT pulse width high	0.45			Ts
t3	FPSHIFT pulse width low	0.45			Ts
t4	data setup to FPSHIFT falling edge	0.45			Ts
t5	data hold from FPSHIFT falling edge	0.45			Ts
t6	FPLINE cycle time	note 2			
t7	FPLINE pulse width low	note 3			
t8	FPFRAME cycle time	note 4			
t9	FPFRAME pulse width low	note 5			
t10	horizontal display period	note 6			
t11	FPLINE setup to FPSHIFT falling edge	0.45			Ts
t12	FPFRAME pulse leading edge to FPLINE pulse leading edge phase difference	note 7			
t13	DRDY to FPSHIFT falling edge setup time	0.45			Ts
t14	DRDY pulse width	note 8			
t15	DRDY falling edge to FPLINE pulse leading edge	note 9			
t16	DRDY hold from FPSHIFT falling edge	0.45			Ts
t17	FPLINE pulse leading edge to DRDY active	note 10		250	Ts

Notes: 1. Ts = pixel clock period = memory clock, [memory clock]/2, [memory clock]/3, [memory clock]/4 (see REG[19h] bits [1:0])

2. $t_{6min} = [((REG[04h] \text{ bits } [6:0]) + 1) * 8 + ((REG[05h] \text{ bits } [4:0]) + 1) * 8] Ts$

3. $t_{7min} = [((REG[07h] \text{ bits } [3:0]) + 1) * 8] Ts$

4. $t_{8min} = [((REG[09h] \text{ bits } [1:0], REG[08h] \text{ bits } [7:0]) + 1) + ((REG[0Ah] \text{ bits } [5:0]) + 1)] \text{ lines}$

5. $t_{9min} = [((REG[0Ch] \text{ bits } [2:0]) + 1)] \text{ lines}$

6. $t_{10min} = [((REG[04h] \text{ bits } [6:0]) + 1) * 8] Ts$

7. $t_{12min} = [((REG[06h] \text{ bits } [4:0]) * 8 + 1) Ts$

8. $t_{14min} = [((REG[04h] \text{ bits } [6:0]) + 1) * 8] Ts$

9. $t_{15min} = [((REG[06h] \text{ bits } [4:0]) + 1) * 8 - 2] Ts$

10. $t_{17min} = [((REG[05h] \text{ bits } [4:0]) + 1) * 8 - ((REG[06h] \text{ bits } [4:0]) + 1) * 8 + 2]$

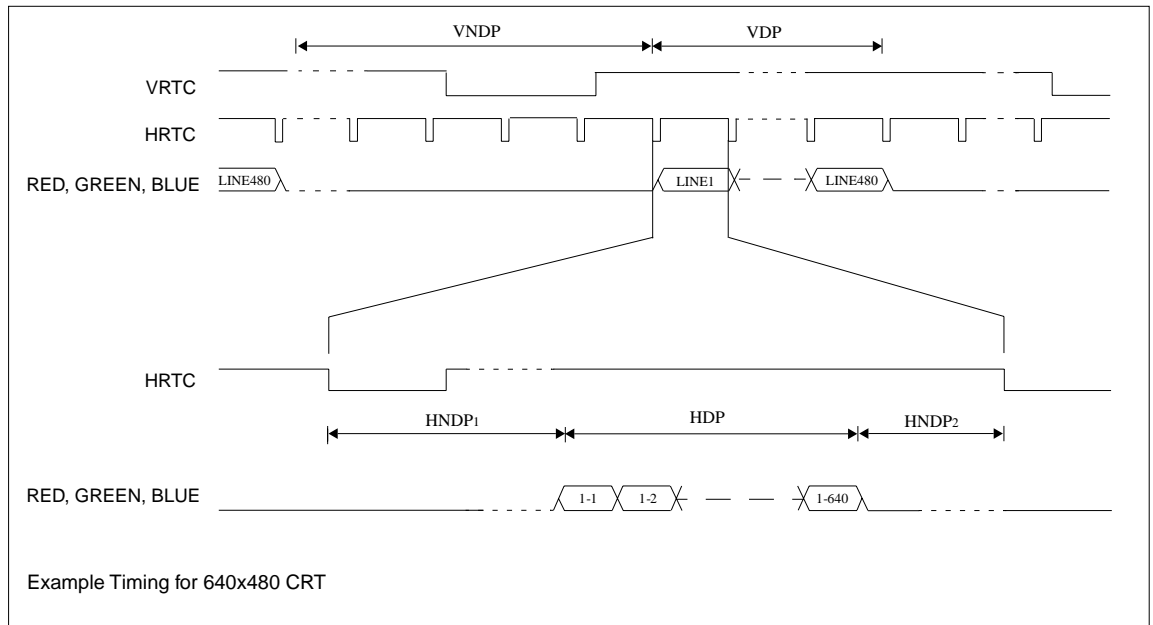
CRT Timing

Figure 7-44 CRT Timing

$VDP = \text{Vertical Display Period} = (\text{REG}[09h] \text{ bits } [1:0], \text{REG}[08h] \text{ bits } [7:0]) + 1$
 $VNDP = \text{Vertical Non-Display Period} = (\text{REG}[0Ah] \text{ bits } [5:0]) + 1$
 $HDP = \text{Horizontal Display Period} = ((\text{REG}[04h] \text{ bits } [6:0]) + 1) * 8T_s$
 $HNDP = \text{Horizontal Non-Display Period} = HNDP_1 + HNDP_2 = ((\text{REG}[05h] \text{ bits } [4:0]) + 1) * 8T_s$

Note: The signals RED, GREEN and BLUE are analog signals from the embedded DAC and represent the color components which make up each pixel.

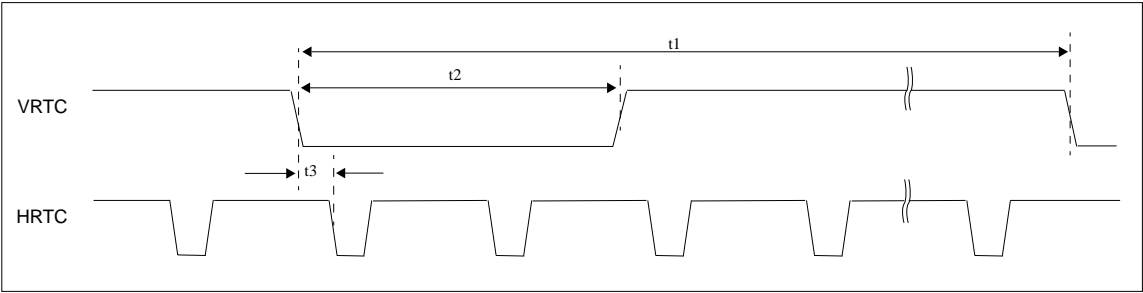


Figure 7-45 CRT A.C. Timing

Table 7-33 CRT A.C. Timing

Symbol	Parameter	Min.	Typ.	Max.	Units
t1	VRTC cycle time		note 1		
t2	VRTC pulse width low		note 2		
t3	VRTC falling edge to FPLINE falling edge phase difference		note 3		

- Notes:** 1. $t1_{min} = [((REG[09h] \text{ bits } 1:0, REG[08h] \text{ bits } 7:0)+1) + ((REG[0Ah] \text{ bits } 6:0)+1)] \text{ lines}$
2. $t2_{min} = [((REG[0Ch] \text{ bits } 2:0)+1)] \text{ lines}$
3. $t3_{min} = [((REG[06h] \text{ bits } 4:0)+1)*8] T_s$

8 REGISTERS

8.1 Register Mapping

The S1D13505 registers are memory mapped. The system addresses the registers through the CS#, M/R#, and AB[5:0] input pins. When CS# = 0 and M/R# = 0, the registers are mapped by address bits AB[5:0], e.g. REG[00h] is mapped to AB[5:0] = 000000, REG[01h] is mapped to AB[5:0] = 000001. See the table below:

Table 8-1 S1D13505 Addressing

CS#	M/R#	Access
0	0	Register access: <ul style="list-style-type: none"> • REG[00h] is addressed when AB[5:0] = 0 • REG[01h] is addressed when AB[5:0] = 1 • REG[n] is addressed when AB[5:0] = n
0	1	Memory access: the 2M byte display buffer is addressed by AB[20:0]
1	×	S1D13505 not selected

8.2 Register Descriptions

Unless specified otherwise, all register bits are reset to 0 during power up. Reserved bits should be written 0 when programming unless otherwise noted.

Revision Code Register

Revision Code Register REG[00h]							RO
Product Code Bit 5	Product Code Bit 4	Product Code Bit 3	Product Code Bit 2	Product Code Bit 1	Product Code Bit 0	Revision Code Bit 1	Revision Code Bit 0

bits 7–2 Product Code Bits [5:0]

This is a read-only register that indicates the product code of the chip.
The product code for the S1D13505F00A is 000011.

bits 1–0 Revision Code Bits [1:0]

This is a read-only register that indicates the revision code of the chip.
The revision code for the S1D13505F00A is 00.

Memory Configuration Registers

Memory Configuration Register REG[01h]							RW
n/a	Refresh Rate Bit 2	Refresh Rate Bit 1	Refresh Rate Bit 0	n/a	WE# Control	n/a	Memory Type

bits 6–4 DRAM Refresh Rate Select Bits [2:0]

These bits specify the divisor used to generate the DRAM refresh rate from the input clock (CLKI).

Table 8-2 DRAM Refresh Rate Selection

DRAM Refresh Rate Select Bits [2:0]	CLKI Frequency Divisor	Example Refresh Rate for CLKI = 33MHz	Example period for 256 refresh cycles at CLKI = 33MHz
000	64	520 kHz	0.5 ms
001	128	260 kHz	1 ms
010	256	130 kHz	2 ms
011	512	65 kHz	4 ms
100	1024	33 kHz	8 ms
101	2048	16 kHz	16 ms
110	4096	8 kHz	32 ms
111	8192	4 kHz	64 ms

bit 2 WE# Control

When this bit = 1, 2-WE# DRAM is selected.
When this bit = 0, 2-CAS# DRAM is selected.

bit 0 Memory Type

When this bit = 1, FPM-DRAM is selected.
When this bit = 0, EDO-DRAM is selected.

This bit should be changed only when there are no read/write DRAM cycles. This condition occurs when all of the following are true: the Display FIFO is disabled (REG[23h] bit 7 = 1), and the Half Frame Buffer is disabled (REG[1Bh] bit 0 = 1), and the Ink/Cursor is inactive (Reg[27h] bits 7-6 = 00). This condition also occurs when the CRT and LCD enable bits (Reg[0Dh] bits 1-0) have remained 0 since chip reset. For further programming information, see “S1D13505 Programming Notes and Examples”, document number X23A-G-003-05.

Panel/Monitor Configuration Registers

Panel Type Register REG[02h]							RW
EL Panel Enable	n/a	Panel Data Width Bit 1	Panel Data Width Bit 0	Panel Data Format Select	Color/Mono Panel Select	Dual/Single Panel Select	TFT/Passive LCD Panel Select

bit 7 EL Panel Mode Enable

When this bit = 1, EL Panel support mode is enabled. Every 262143 frames (approximately 1 hour at 60Hz frame rate) the identical panel data is sent to two consecutive frames, i.e. the frame rate modulation circuitry is frozen for one frame.

bits 5–4 Panel Data Width Bits [1:0]

These bits select the LCD interface data width as shown in the following table.

Table 8-3 Panel Data Width Selection

Panel Data Width Bits [1:0]	Passive LCD Panel Data Width Size	TFT/D-TFD Panel Data Width Size
00	4-bit	9-bit
01	8-bit	12-bit
10	16-bit	16-bit
11	Reserved	Reserved

bit 3 Panel Data Format Select

When this bit = 1, color passive LCD panel data format 2 is selected.

When this bit = 0, passive LCD panel data format 1 is selected.

bit 2 Color/Mono Panel Select

When this bit = 1, color passive LCD panel is selected.

When this bit = 0, monochrome passive LCD panel is selected.

bit 1 Dual/Single Panel Select

When this bit = 1, dual passive LCD panel is selected.

When this bit = 0, single passive LCD panel is selected.

bit 0 TFT/Passive LCD Panel Select

When this bit = 1, TFT/D-TFD panel is selected.

When this bit = 0, passive LCD panel is selected.

MOD Rate Register REG[03h]							RW
n/a	n/a	MOD Rate Bit 5	MOD Rate Bit 4	MOD Rate Bit 3	MOD Rate Bit 2	MOD Rate Bit 1	MOD Rate Bit 0

bits 5–0 MOD Rate Bits [5:0]

When the DRDY pin is configured as MOD, this register controls the toggle rate of the MOD output. When this register is zero, the MOD output signal toggles every FPFREAME. When this register is non-zero, its value represents the number of FPLINE pulses between toggles of the MOD output signal.

Horizontal Display Width Register

REG[04h]

RW

n/a	Horizontal Display Width Bit 6	Horizontal Display Width Bit 5	Horizontal Display Width Bit 4	Horizontal Display Width Bit 3	Horizontal Display Width Bit 2	Horizontal Display Width Bit 1	Horizontal Display Width Bit 0
-----	--------------------------------	--------------------------------	--------------------------------	--------------------------------	--------------------------------	--------------------------------	--------------------------------

bits 6–0 Horizontal Display Width Bits [6:0]

These bits specify the horizontal display width.

Horizontal display width (pixels) = (Horizontal Display Width Bits [6:0] + 1) × 8

The maximum horizontal display width is 1024 pixels.

- Notes:**
- This register must be programmed such that REG[04h] ≥ 3 (32 pixels).
 - When setting a horizontal resolution greater than 767 pixels, with a color depth of 15/16 bpp, the Memory Offset Registers (REG[16h], REG[17h]) must be set to a virtual horizontal pixelresolution of 1024.

Horizontal Non-Display Period Register

REG[05h]

RW

n/a	n/a	n/a	Horizontal Non-Display Period Bit 4	Horizontal Non-Display Period Bit 3	Horizontal Non-Display Period Bit 2	Horizontal Non-Display Period Bit 1	Horizontal Non-Display Period Bit 0
-----	-----	-----	-------------------------------------	-------------------------------------	-------------------------------------	-------------------------------------	-------------------------------------

bits 4–0 Horizontal Non-Display Period Bits [4:0]

These bits specify the horizontal non-display period.

Horizontal non-display period (pixels) = (Horizontal Non-Display Period Bits [4:0] + 1) × 8

The recommended minimum value which should be programmed into this register is 3 (32 pixels). The maximum value which can be programmed into this register is 1Fh, which gives a horizontal non-display period of 256 pixels.

- Note:** This register must be programmed such that
 REG[05h] ≥ 3 and (REG[05h] + 1) ≥ (REG[06h] + 1) + (REG[07h] bits [3:0] + 1)

HRTC/FPLINE Start Position Register

REG[06h]

RW

n/a	n/a	n/a	HRTC/FPLINE Start Position Bit 4	HRTC/FPLINE Start Position Bit 3	HRTC/FPLINE Start Position Bit 2	HRTC/FPLINE Start Position Bit 1	HRTC/FPLINE Start Position Bit 0
-----	-----	-----	----------------------------------	----------------------------------	----------------------------------	----------------------------------	----------------------------------

bits 4–0 HRTC/FPLINE Start Position Bits [4:0]

For CRT and TFT/D-TFD, these bits specify the delay from the start of the horizontal non-display period to the leading edge of the HRTC pulse and FPLINE pulse respectively.

HRTC/FPLINE start position (pixels) = (HRTC/FPLINE Start Position Bits [4:0] + 1) × 8 - 2

- Note:** This register must be programmed such that
 (REG[05h] + 1) ≥ (REG[06h] + 1) + (REG[07h] bits [3:0] + 1)

HRTC/FPLINE Pulse Width Register REG[07h]								RW
HRTC Polarity Select	FPLINE Polarity Select	n/a	n/a	HRTC/FPLINE Pulse Width Bit 3	HRTC/FPLINE Pulse Width Bit 2	HRTC/FPLINE Pulse Width Bit 1	HRTC/FPLINE Pulse Width Bit 0	

bit 7 HRTC Polarity Select

This bit selects the polarity of the HRTC pulse to the CRT.

When this bit = 1, the HRTC pulse is active high. When this bit = 0, the HRTC pulse is active low.

bit 6 FPLINE Polarity Select

This bit selects the polarity of the FPLINE pulse to TFT/D-TFD or passive LCD.

When this bit = 1, the FPLINE pulse is active high for TFT/D-TFD and active low for passive LCD. When this bit = 0, the FPLINE pulse is active low for TFT/D-TFD and active high for passive LCD.

Table 8-4 FPLINE Polarity Selection

FPLINE Polarity Select	Passive LCD FPLINE Polarity	TFT/D-TFD FPLINE Polarity
0	active high	active low
1	active low	active high

bits 3–0 HRTC/FPLINE Pulse Width Bits [3:0]

For CRT and TFT/D-TFD, these bits specify the pulse width of HRTC and FPLINE respectively. For passive LCD, FPLINE is automatically created and these bits have no effect.

HRTC/FPLINE pulse width (pixels) = (HRTC/FPLINE Pulse Width Bits [3:0] + 1) × 8

The maximum HRTC pulse width is 128 pixels.

Note: This register must be programmed such that
 $(\text{REG}[05\text{h}] + 1) \geq (\text{REG}[06\text{h}] + 1) + (\text{REG}[07\text{h}] \text{ bits } [3:0] + 1)$

Vertical Display Height Register 0 REG[08h]								RW
Vertical Dis- play Height Bit 7	Vertical Dis- play Height Bit 6	Vertical Dis- play Height Bit 5	Vertical Dis- play Height Bit 4	Vertical Dis- play Height Bit 3	Vertical Dis- play Height Bit 2	Vertical Dis- play Height Bit 1	Vertical Dis- play Height Bit 0	

Vertical Display Height Register 1 REG[09h]								RW
n/a	n/a	n/a	n/a	n/a	n/a	Vertical Dis- play Height Bit 9	Vertical Dis- play Height Bit 8	

REG[08h] bits 7–0 Vertical Display Height Bits [9:0]**REG[09h] bits 1–0 These bits specify the vertical display height.**

Vertical display height (lines) = Vertical Display Height Bits [9:0] + 1

- For CRT, TFT/D-TFD, and single passive LCD panel this register is programmed to:
 (vertical resolution of the display) - 1, e.g. EFh for a 240-line display.
- For dual-panel passive LCD not in simultaneous display mode, this register is programmed to:
 ((vertical resolution of the display)/2) - 1, e.g. EFh for a 480-line display.
- For all simultaneous display modes, this register is programmed to:
 (vertical resolution of the CRT) - 1, e.g. 1DFh for a 480-line CRT.

Vertical Non-Display Period Register

REG[0Ah]

RW

Vertical Non-Display Period Status (RO)	n/a	Vertical Non-Display Period Bit 5	Vertical Non-Display Period Bit 4	Vertical Non-Display Period Bit 3	Vertical Non-Display Period Bit 2	Vertical Non-Display Period Bit 1	Vertical Non-Display Period Bit 0
---	-----	-----------------------------------	-----------------------------------	-----------------------------------	-----------------------------------	-----------------------------------	-----------------------------------

bit 7 Vertical Non-Display Period Status

This is a read-only status bit.

When this bit = 1, a vertical non-display period is indicated.

When this bit = 0, a vertical display period is indicated.

bits 5–0 Vertical Non-Display Period Bits [5:0]

These bits specify the vertical non-display period.

Vertical non-display period (lines) = Vertical Non-Display Period Bits [5:0]

Note: This register must be programmed such that
 $\text{REG}[0Ah] \geq 1$ and $(\text{REG}[0Ah] \text{ bits } [5:0] + 1) \geq (\text{REG}[0Bh] + 1) + (\text{REG}[0Ch] \text{ bits } [2:0] + 1)$

VRTC/FPFRAME Start Position Register

REG[0Bh]

RW

n/a	n/a	VRTC/FPFRAME Start Position Bit 5	VRTC/FPFRAME Start Position Bit 4	VRTC/FPFRAME Start Position Bit 3	VRTC/FPFRAME Start Position Bit 2	VRTC/FPFRAME Start Position Bit 1	VRTC/FPFRAME Start Position Bit 0
-----	-----	-----------------------------------	-----------------------------------	-----------------------------------	-----------------------------------	-----------------------------------	-----------------------------------

bits 5–0 VRTC/FPFRAME Start Position Bits [5:0]

For CRT and TFT/D-TFD, these bits specify the delay in lines from the start of the vertical non-display period to the leading edge of the VRTC pulse and FPFRAME pulse respectively. For passive LCD, FPFRAME is automatically created and these bits have no effect.

VRTC/FPFRAME start position (lines) = VRTC/FPFRAME Start Position Bits [5:0] + 1

The maximum start delay is 64 lines.

Note: This register must be programmed such that
 $(\text{REG}[0Ah] \text{ bits } [5:0] + 1) \geq (\text{REG}[0Bh] + 1) + (\text{REG}[0Ch] \text{ bits } [2:0] + 1)$
 For exact timing please use the timing diagrams in section 7.5

VRTC/FPFRAME Pulse Width Register							RW
REG[0Ch]							
VRTC Polarity Select	FPFRAME Polarity Select	n/a	n/a	n/a	VRTC/FPFRAME Pulse Width Bit 2	VRTC/FPFRAME Pulse Width Bit 1	VRTC/FPFRAME Pulse Width Bit 0

bit 7 VRTC Polarity Select

This bit selects the polarity of the VRTC pulse to the CRT.

When this bit = 1, the VRTC pulse is active high.

When this bit = 0, the VRTC pulse is active low.

bit 6 FPFRAME Polarity Select

This bit selects the polarity of the FPFRAME pulse to the TFT/D-TFD or passive LCD.

When this bit = 1, the FPFRAME pulse is active high for TFT/D-TFD and active low for passive.

When this bit = 0, the FPFRAME pulse is active low for TFT/D-TFD and active high for passive.

Table 8-5 FPFRAME Polarity Selection

FPFRAME Polarity Select	Passive LCD FPFRAME Polarity	TFT/D-TFD FPFRAME Polarity
0	active high	active low
1	active low	active high

bits 2–0 VRTC/FPFRAME Pulse Width Bits [2:0]

For CRT and TFT/D-TFD, these bits specify the pulse width of VRTC and FPFRAME respectively. For passive LCD, FPFRAME is automatically created and these bits have no effect.

$\text{VRTC/FPFRAME pulse width (lines)} = \text{VRTC/FPFRAME Pulse Width Bits [2:0]} + 1$

Note: This register must be programmed such that
 $(\text{REG}[0Ah] \text{ bits } [5:0] + 1) \geq (\text{REG}[0Bh] + 1) + (\text{REG}[0Ch] \text{ bits } [2:0] + 1)$

Display Configuration Registers

Display Mode Register REG[0Dh]							RW
SwivelView™ Enable	Simultaneous Display Option Select Bit 1	Simultaneous Display Option Select Bit 0	Bit-per-pixel Select Bit 2	Bit-per-pixel Select Bit 1	Bit-per-pixel Select Bit 0	CRT Enable	LCD Enable

bit 7 SwivelView™ Enable

When this bit = 1, all CPU accesses to the display buffer are translated to provide clockwise 90° hardware rotation of the display image. Refer to Section 13, “SwivelView™” for application and limitations.

bits 6–5 Simultaneous Display Option Select Bits [1:0]

These bits are used to select one of four different simultaneous display mode options: Normal, Line Doubling, Interlace, or Even Scan Only. The purpose of these modes is to manipulate the vertical resolution of the image so that it fits on both the CRT, typically 640x480, and LCD. The following table describes the four modes using a 640x480 CRT as an example:

Table 8-6 Simultaneous Display Option Selection

Simultaneous Display Option Select Bits [1:0]	Simultaneous Display Mode	Mode Description
00	Normal	The image is not manipulated. This mode is used when the CRT and LCD have the same resolution, e.g. 480 lines. It is necessary to suit the vertical retrace period to the CRT. This results in a lower LCD duty cycle (1/525 compared to the usual 1/481). This reduced duty cycle may result in lower contrast on the LCD.
01	Line Doubling	Each line is replicated on the CRT. This mode is used to display a 240-line image on a 240-line LCD and stretch it to a 480-line image on the CRT. The CRT has a heightened aspect ratio. It is necessary to suit the vertical retrace period to the CRT. This results in a lower LCD duty cycle (2/525 compared to the usual 1/241). This reduced duty cycle is not extreme and the contrast of the LCD image should not be greatly reduced.
10	Interlace	The odd and even fields of a 480-line image are interlaced on the LCD. This mode is used to display a 480-line image on the CRT and squash it onto a 240-line LCD. The full image is viewed on the LCD but the interlacing may create flicker. The LCD has a shortened aspect ratio. It is necessary to suit the vertical retrace period to the CRT. This results in a lower LCD duty cycle (2/525 compared to the usual 1/241). This reduced duty cycle is not extreme and the contrast of the LCD image should not be greatly reduced.
11	Even Scan Only	Only the even field of a 480-line image is displayed on the LCD. This is an alternate method to display a 480-line image on the CRT and squash it onto a 240-line LCD. Only the even scans are viewed on the LCD. The LCD has a shortened aspect ratio. It is necessary to suit the vertical retrace period to the CRT. This results in a lower LCD duty cycle (2/525 compared to the usual 1/241). This reduced duty cycle is not extreme and the contrast of the LCD image should not be greatly reduced.

Notes: 1. Dual Panel Considerations:

When configured for a dual LCD panel and using Simultaneous Display, the Half Frame Buffer Disable, REG[1Bh] bit 0, must be set to 1.

This results in a lower contrast on the LCD panel, which may require adjustment.

- The Line doubling option is not supported with dual panel.

bits 4–2 Bit-Per-Pixel Select Bits [2:0]

These bits select the color depth (bpp) for the displayed data. See Section 10.1, “*Display Mode Data Format*” for details of how the pixels are mapped into the image buffer.

Table 8-7 Bits-Per-Pixel Selection

Bit-per-pixel Select Bits [2:0]	Color Depth (bpp)
000	1 bpp
001	2 bpp
010	4 bpp
011	8 bpp
100	15 bpp
101	16 bpp
110 – 111	Reserved

bit 1 CRT Enable

This bit enables the CRT monitor.

When this bit = 1, the CRT is enabled.

When this bit = 0, the CRT is disabled.

bit 0 LCD Enable

This bit enables the LCD panel.

Programming this bit from a 0 to a 1 starts the LCD power-on sequence.

Programming this bit from a 1 to a 0 starts the LCD power-off sequence.

Screen 1 Line Compare Register 0

REG[0Eh]							RW
Screen 1 Line Compare Bit 7	Screen 1 Line Compare Bit 6	Screen 1 Line Compare Bit 5	Screen 1 Line Compare Bit 4	Screen 1 Line Compare Bit 3	Screen 1 Line Compare Bit 2	Screen 1 Line Compare Bit 1	Screen 1 Line Compare Bit 0

Screen 1 Line Compare Register 1

REG[0Fh]							RW
n/a	n/a	n/a	n/a	n/a	n/a	Screen 1 Line Compare Bit 9	Screen 1 Line Compare Bit 8

REG[0Eh] bits 7–0 Screen 1 Line Compare Bits [9:0]**REG[0Fh] bits 1–0 These bits are set to 1 during power-on.**

The display can be split into two images: Screen 1 and Screen 2, with Screen 1 above Screen 2. This 10-bit value specifies the height of Screen 1.

Height of Screen 1 (lines) = Screen 1 Line Compare Bits [9:0] + 1

If the height of Screen 1 is less than the display height then the remainder of the display is taken up by Screen 2. For normal operation (no split screen) this register must be set greater than the Vertical Display Height register (e.g. set to the reset value of 3FFh).

See Section 10, “*Display Configuration*” for details.

Screen 1 Display Start Address Register 0

REG[10h]							RW
Start Address Bit 7	Start Address Bit 6	Start Address Bit 5	Start Address Bit 4	Start Address Bit 3	Start Address Bit 2	Start Address Bit 1	Start Address Bit 0

Screen 1 Display Start Address Register 1

REG[11h]							RW
Start Address Bit 15	Start Address Bit 14	Start Address Bit 13	Start Address Bit 12	Start Address Bit 11	Start Address Bit 10	Start Address Bit 9	Start Address Bit 8

Screen 1 Display Start Address Register 2

REG[12h]							RW
n/a	n/a	n/a	n/a	Start Address Bit 19	Start Address Bit 18	Start Address Bit 17	Start Address Bit 16

REG[10h] bits 7–0 Screen 1 Start Address Bits [19:0]

REG[11h] bits 7–0 These registers form the 20-bit address for the starting word of the Screen 1 image in the display buffer.

REG[12h] bits 3–0

Note that this is a word address. A combination of this register and the Pixel Panning register (REG[18h]) can be used to uniquely identify the start (top left) pixel within the Screen 1 image stored in the display buffer.

See Section 10, “*Display Configuration*” for details.

Screen 2 Display Start Address Register 0

REG[13h]							RW
Start Address Bit 7	Start Address Bit 6	Start Address Bit 5	Start Address Bit 4	Start Address Bit 3	Start Address Bit 2	Start Address Bit 1	Start Address Bit 0

Screen 2 Display Start Address Register 1

REG[14h]							RW
Start Address Bit 15	Start Address Bit 14	Start Address Bit 13	Start Address Bit 12	Start Address Bit 11	Start Address Bit 10	Start Address Bit 9	Start Address Bit 8

Screen 2 Display Start Address Register 2

REG[15h]							RW
n/a	n/a	n/a	n/a	Start Address Bit 19	Start Address Bit 18	Start Address Bit 17	Start Address Bit 16

REG[13h] bits 7–0 Screen 2 Start Address Bits [19:0]

REG[14h] bits 7–0 These registers form the 20-bit address for the starting word of the Screen 2 image in the display buffer.

REG[15h] bits 3–0

Note that this is a word address.

A combination of this register and the Pixel Panning register (REG[18h]) can be used to uniquely identify the start (top left) pixel within the Screen 2 image stored in the display buffer.

See Section 10, “*Display Configuration*” for details.

Memory Address Offset Register 0

REG[16h]							RW
Memory Address Offset Bit 7	Memory Address Offset Bit 6	Memory Address Offset Bit 5	Memory Address Offset Bit 4	Memory Address Offset Bit 3	Memory Address Offset Bit 2	Memory Address Offset Bit 1	Memory Address Offset Bit 0

Memory Address Offset Register 1

REG[17h]							RW
n/a	n/a	n/a	n/a	n/a	Memory Address Offset Bit 10	Memory Address Offset Bit 9	Memory Address Offset Bit 8

REG[16h] bits 7–0 Memory Address Offset Bits [10:0]

REG[17h] bits 2–0 These bits form the 11-bit address offset from the starting word of line *n* to the starting word of line *n*+1. This value is applied to both Screen 1 and Screen 2.

Note that this value is in words.

A virtual image can be formed by setting this register to a value greater than the width of the display. The displayed image is a window into the larger virtual image.

See Section 10, “*Display Configuration*” for details.

Pixel Panning Register							RW
Screen 2 Pixel Panning Bit 3	Screen 2 Pixel Panning Bit 2	Screen 2 Pixel Panning Bit 1	Screen 2 Pixel Panning Bit 0	Screen 1 Pixel Panning Bit 3	Screen 1 Pixel Panning Bit 2	Screen 1 Pixel Panning Bit 1	Screen 1 Pixel Panning Bit 0

This register is used to control the horizontal pixel panning of Screen 1 and Screen 2. Each screen can be independently panned to the left by programming its respective Pixel Panning Bits to a non-zero value. The value represents the number of pixels panned. The maximum pan value is dependent on the display mode.

Table 8-8 Pixel Panning Selection

Display Mode	Maximum Pan Value	Pixel Panning Bits Active
1 bpp	16	Bits [3:0]
2 bpp	8	Bits [2:0]
4 bpp	4	Bits [1:0]
8 bpp	1	Bit 0
15/16 bpp	0	none

Smooth horizontal panning can be achieved by a combination of this register and the Display Start Address registers.

See Section 10, “*Display Configuration*” for details.

bits 7–4 Screen 2 Pixel Panning Bits [3:0]

Pixel panning bits for screen 2.

bits 3–0 Screen 1 Pixel Panning Bits [3:0]

Pixel panning bits for screen 1.

Clock Configuration Register

Clock Configuration Register REG[19h]							RW
Reserved	n/a	n/a	n/a	n/a	MCLK Divide Select	PCLK Divide Select Bit 1	PCLK Divide Select Bit 0

bit 7 Reserved

This bit must be set to 0.

bit 2 MCLK Divide Select

When this bit = 1 the MCLK frequency is half of its source frequency.

When this bit = 0 the MCLK frequency is equal to its source frequency.

The MCLK frequency should always be set to the maximum frequency allowed by the DRAM; this provides maximum performance and minimum overall system power consumption.

bits 1–0 PCLK Divide Select Bits [1:0]

These bits select the MCLK: PCLK frequency ratio.

Table 8-9 PCLK Divide Selection

PCLK Divide Select Bits [1:0]	MCLK : PCLK Frequency Ratio
00	1 : 1
01	2 : 1
10	3 : 1
11	4 : 1

See Section on “*Maximum MCLK: PCLK Ratios*” for selection of clock ratios.

Power Save Configuration Registers

Power Save Configuration Register							RW
REG[1Ah]							
Power Save Status RO	n/a	n/a	n/a	LCD Power Disable	Suspend Refresh Select Bit 1	Suspend Refresh Select Bit 0	Software Suspend Mode Enable

bit 7 Power Save Status

This is a read-only status bit.

This bit indicates the power-save state of the chip.

When this bit = 1, the panel has been powered down and the memory controller is either in self refresh mode or is performing only CAS-before-RAS refresh cycles.

When this bit = 0, the chip is either powered up, in transition of powering up, or in transition of powering down. See Section 15, “Power Save Modes” for details.

bit 3 LCD Power Disable

This bit is used to override the panel on/off sequencing logic.

When this bit = 0 the LCDPWR output is controlled by the panel on/off sequencing logic.

When this bit = 1 the LCDPWR output is directly forced to the off state.

The LCDPWR “On/Off” polarity is configured by MD10 at the rising edge of RESET# (MD10 = 0 configures LCDPWR = 0 as the Off state; MD10 = 1 configures LCDPWR = 1 as the Off state).

bits 2–1 Suspend Refresh Select Bits [1:0]

These bits specify the type of DRAM refresh to use in Suspend mode.

Table 8-10 Suspend Refresh Selection

Suspend Refresh Select Bits [1:0]	DRAM Refresh Type
00	CAS-before-RAS (CBR) refresh
01	Self-Refresh
1x	No Refresh

Note: These bits should not be changed when suspend mode is active.

bit 0 Software Suspend Mode Enable

When this bit = 1 software Suspend mode is enabled.

When this bit = 0 software Suspend mode is disabled.

See Section 15, “Power Save Modes” for details.

Miscellaneous Registers

Miscellaneous Disable Register							
REG[1Bh]							RW
Host Interface Disable	n/a	n/a	n/a	n/a	n/a	n/a	Half Frame Buffer Disable

bit 7 Host Interface Disable

This bit is set to 1 during power-on/reset.

This bit must be programmed to 0 to enable the Host Interface. When this bit is high, all memory and all registers except REG[1Ah] (read-only) and REG[1Bh] are inaccessible.

bit 0 Half Frame Buffer Disable

This bit is used to disable the Half Frame Buffer.

When this bit = 1, the Half Frame Buffer is disabled.

When this bit = 0, the Half Frame Buffer is enabled.

When a single panel is selected, the Half Frame Buffer is automatically disabled and this bit has no effect.

The half frame buffer is needed to fully support dual panels. Disabling the Half Frame Buffer reduces memory bandwidth requirements and increases the supportable pixel clock frequency, but results in reduced contrast on the LCD panel (the duty cycle of the LCD is halved). This mode is not normally used except under special circumstances such as simultaneous display on a CRT and dual panel LCD. When this mode is used the Alternate Frame Rate Modulation scheme should be used (see REG[31h]). For details on Frame Rate calculation see Section 14.2, “Frame Rate Calculation”.

MD Configuration Readback Register 0							
REG[1Ch]							RO
MD[7] Status	MD[6] Status	MD[5] Status	MD[4] Status	MD[3] Status	MD[2] Status	MD[1] Status	MD[0] Status

MD Configuration Readback Register 1							
REG[1Dh]							RO
MD[15] Status	MD[14] Status	MD[13] Status	MD[12] Status	MD[11] Status	MD[10] Status	MD[9] Status	MD[8] Status

REG[1Ch] bits 7–0 MD[15:0] Configuration Status

REG[1Dh] bits 7–0 These are read-only status bits for the MD[15:0] pins configuration status at the rising edge of RESET#. MD[15:0] are used to configure the chip at the rising edge of RESET# – see “Pin Descriptions and Summary of Configuration Options” for details.

General IO Pins Configuration Register 0

REG[1Eh]							RO
n/a	n/a	n/a	n/a	GPIO3 Pin IO Config.	GPIO2 Pin IO Config.	GPIO1 Pin IO Config.	n/a

Pins MA9, MA10, MA11 are multi-functional – they can be DRAM address outputs or general purpose IO dependent on the DRAM type. MD[7:6] are used to identify the DRAM type and configure these pins as follows:

Table 8-11 MA/GPIO Pin Functionality

MD[7:6] at Rising Edge of RESET#	Pin Function		
	MA9	MA10	MA11
00	GPIO3	GPIO1	GPIO2
01	MA9	GPIO1	GPIO2
10	MA9	GPIO1	GPIO2
11	MA9	MA10	MA11

These bits are used to control the direction of these pins when they are used as general purpose IO. These bits have no effect when the pins are used as DRAM address outputs.

bit 3 GPIO3 Pin IO Configuration

When this bit = 1, the GPIO3 pin is configured as an output pin.

When this bit = 0 (default), the GPIO3 pin is configured as an input pin.

bit 2 GPIO2 Pin IO Configuration

When this bit = 1, the GPIO2 pin is configured as an output pin.

When this bit = 0 (default), the GPIO2 pin is configured as an input pin.

bit 1 GPIO1 Pin IO Configuration

When this bit = 1, the GPIO1 pin is configured as an output pin.

When this bit = 0 (default), the GPIO1 pin is configured as an input pin.

General IO Pins Configuration Register 1

REG[1Fh]							RW
n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a

This register position is reserved for future use.

General IO Pins Control Register 0

REG[20h]							RW
n/a	n/a	n/a	n/a	GPIO3 Pin IO Status	GPIO2 Pin IO Status	GPIO1 Pin IO Status	n/a

bit 3 GPIO3 Pin IO Status

When GPIO3 is configured as an output (see REG[1Eh]), a “1” in this bit drives GPIO3 high and a “0” in this bit drives GPIO3 low.

When GPIO3 is configured as an input, a read from this bit returns the status of GPIO3.

bit 2 GPIO2 Pin IO Status

When GPIO2 is configured as an output (see REG[1Eh]), a “1” in this bit drives GPIO2 high and a “0” in this bit drives GPIO2 low.

When GPIO2 is configured as an input, a read from this bit returns the status of GPIO2.

bit 1 GPIO1 Pin IO Status

When GPIO1 is configured as an output (see REG[1Eh]), a “1” in this bit drives GPIO1 high and a “0” in this bit drives GPIO1 low.

When GPIO1 is configured as an input, a read from this bit returns the status of GPIO1.

GPIO Status / Control Register 1

REG[21h]

RW

GPO Control	n/a	n/a	n/a	n/a	n/a	n/a	n/a
-------------	-----	-----	-----	-----	-----	-----	-----

bit 7 GPO Control

This bit is used to control the state of the SUSPEND# when it is configured as GPO. The SUSPEND# pin can be used as a power-down input (SUSPEND#) or as an output (GPO) possibly used for controlling the LCD backlight power:

- When MD9 = 0 at rising edge of RESET#, SUSPEND#/GPO is an active-low Schmitt input used to put the S1D13505 into suspend mode – see “*Power Save Modes*” for details.
- When MD[10:9] = 01 at rising edge of RESET#, SUSPEND#/GPO is an output with a reset state of 0.
- When MD[10:9] = 11 at rising edge of RESET#, SUSPEND#/GPO is an output with a reset state of 1.

When this bit = 1 the GPO output is set to the reset state.

When this bit = 0 the GPO output pin is set to the inverse of the reset state.

Performance Enhancement Register 0

REG[22h]

RW

Reserved	RC Timing Value Bit 1	RC Timing Value Bit 0	RAS#-to-CAS# Delay Value	RAS# Pre-charge Timing Value Bit 1	RAS# Pre-charge Timing Value Bit 0	Reserved	Reserved
----------	-----------------------	-----------------------	--------------------------	------------------------------------	------------------------------------	----------	----------

Note: Changing this register to non-zero value, or to a different non-zero value, should be done only when there are no read/write DRAM cycles. This condition occurs when all of the following are true: the Display FIFO is disabled (REG[23h] bit 7 = 1), and the Half Frame Buffer is disabled (REG[1Bh] bit 0 = 1), and the Ink/Cursor is inactive (Reg[27h] bits 7-6 = 00). This condition also occurs when the CRT and LCD enable bits (Reg[0Dh] bits 1-0) have remained 0 since chip reset. For further programming information, see “*S1D13505 Programming Notes and Examples*”, document number X23A-G-003-05.

bit 7 Reserved**bits 6–5 RC Timing Value (NRC) Bits [1:0]**

These bits select the DRAM random-cycle timing parameter, trc. These bits specify the number (NRC) of MCLK periods (T_M) used to create trc. NRC should be chosen to meet trc as well as t_{RAS}, the RAS pulse width. Use the following two formulae to calculate NRC then choose the larger value. Note, these formulae assume an MCLK duty cycle of 50 +/- 5%.

$$NRC = \text{Round-Up} (trc/T_M)$$

$$NRC = \text{Round-Up} (t_{RAS}/T_M + NRP) \text{ if } NRP = 1 \text{ or } 2$$

$$= \text{Round-Up} (t_{RAS}/T_M + 1.55) \text{ if } NRP = 1.5$$

The resulting trc is related to NRC as follows:

$$trc = (NRC) T_M$$

Table 8-12 Minimum Memory Timing Selection

REG[22h] Bits [6:5]	NRC	Minimum Random Cycle Width (trc)
00	5	5 T _M
01	4	4 T _M
10	3	3 T _M
11	Reserved	Reserved

bit 4 RAS#-to-CAS# Delay Value (NRCD)

This bit selects the DRAM RAS#-to-CAS# delay parameter, trCD. This bit specifies the number (NRCD) of MCLK periods (T_M) used to create trCD. NRCD must be chosen to satisfy the RAS# access time, trAC. Note, these formulae assume an MCLK duty cycle of 50 ± 5%.

$$\begin{aligned} \text{NRCD} &= \text{Round-Up} ((\text{trAC} + 5)/\text{T}_M - 1) && \text{if EDO and NRP} = 1 \text{ or } 2 \\ &= 2 && \text{if EDO and NRP} = 1.5 \\ &= \text{Round-Up} (\text{trAC}/\text{T}_M - 1) && \text{if FPM and NRP} = 1 \text{ or } 2 \\ &= \text{Round-Up} (\text{trAC}/\text{T}_M - 0.45) && \text{if FPM and NRP} = 1.5 \end{aligned}$$

Note that for EDO-DRAM and NRP = 1.5, this bit is automatically forced to 0 to select 2 MCLK for NRCD. This is done to satisfy the CAS# address setup time, tASC.

The resulting trCD is related to NRCD as follows:

$$\begin{aligned} \text{trCD} &= (\text{NRCD}) \text{ T}_M && \text{if EDO and NRP} = 1 \text{ or } 2 \\ \text{trCD} &= (1.5) \text{ T}_M && \text{if EDO and NRP} = 1.5 \\ \text{trCD} &= (\text{NRCD} + 0.5) \text{ T}_M && \text{if FPM and NRP} = 1 \text{ or } 2 \\ \text{trCD} &= (\text{NRCD}) \text{ T}_M && \text{if FPM and NRP} = 1.5 \end{aligned}$$

Table 8-13 RAS#-to-CAS# Delay Timing Select

REG[22h] Bit 4	NRCD	RAS#-to-CAS# Delay (trCD)
0	2	2
1	1	1

bits 3–2 RAS# Precharge Timing Value (NRP) Bits [1:0]

Minimum Memory Timing for RAS# precharge

These bits select the DRAM RAS# Precharge timing parameter, trP. These bits specify the number (NRP) of MCLK periods (T_M) used to create trP – see the following formulae. Note, these formulae assume an MCLK duty cycle of 50 +/- 5%.

$$\begin{aligned} \text{NRP} &= 1 && \text{if } (\text{trP}/\text{T}_M) < 1 \\ &= 1.5 && \text{if } 1 \leq (\text{trP}/\text{T}_M) < 1.45 \\ &= 2 && \text{if } (\text{trP}/\text{T}_M) \geq 1.45 \end{aligned}$$

The resulting trP is related to NRP as follows:

$$\begin{aligned} \text{trP} &= (\text{NRP} + 0.5) \text{ T}_M && \text{if FPM refresh cycle and NRP} = 1 \text{ or } 2 \\ \text{trP} &= (\text{NRP}) \text{ T}_M && \text{for all other} \end{aligned}$$

bits 1–0 Reserved

These bits must be set to 0.

Table 8-14 RAS# Precharge Timing Select

REG[22h] Bits [3:2]	NRP	RAS# Precharge Width (trP)
00	2	2
01	1.5	1.5
10	1	1
11	Reserved	Reserved

Optimal DRAM Timing

The following table contains the optimally programmed values of NRC, NRP, and NRCD for different DRAM types, at maximum MCLK frequencies.

Table 8-15 Optimal NRC, NRP, and NRCD Values at Maximum MCLK Frequency

DRAM Type	DRAM Speed (ns)	T _M (ns)	NRC (#MCLK)	NRP (#MCLK)	NRCD (#MCLK)
EDO	50	25	4	1.5	2
	60	30	4	1.5	2
	70	33	5	2	2
FPM	60	40	4	1.5	2
	70	50	3	1.5	1

bits 1–0 Reserved

This reserved bit must be set to 0.

Performance Enhancement Register 1

REG[23h]

RW

Display FIFO Disable	CPU to Memory Wait State Bit 1	CPU to Memory Wait State Bit 0	Display FIFO Threshold Bit 4	Display FIFO Threshold Bit 3	Display FIFO Threshold Bit 2	Display FIFO Threshold Bit 1	Display FIFO Threshold Bit 0
----------------------	--------------------------------	--------------------------------	------------------------------	------------------------------	------------------------------	------------------------------	------------------------------

bit 7 Display FIFO Disable

When this bit = 1 the display FIFO is disabled and all data outputs are forced to zero (i.e., the screen is blanked). This accelerates screen updates by allocating more memory bandwidth to CPU accesses.

When this bit = 0 the display FIFO is enabled.

Note: For further performance increase in dual panel mode disable the half frame buffer (see “Miscellaneous Registers”) and disable the cursor (see “Ink/Cursor Registers”).

bits 6–5 CPU to Memory Wait State Bits [1:0]

These bits are used to optimize the handshaking between the host interface and the memory controller. The bits should be set according to the relationship between BCLK and MCLK – see the table below where T_B and T_M are the BCLK and MCLK periods respectively.

Table 8-16 Minimum Memory Timing Selection

Wait State Bits [1:0]	Condition
00	no restrictions (default)
01	2T _M - 4ns > T _B
10	undefined
11	undefined

bits 4–0 Display FIFO Threshold Bits [4:0]

These bits specify the display FIFO depth required to sustain uninterrupted display fetches. When these bits are all 0s, the display FIFO depth is calculated automatically. These bits should always be set to 0, except in the following configurations:

Landscape mode at 15/16 bpp (with MCLK=PCLK),

Portrait mode at 8/16 bpp (with MCLK=PCLK).

When in the above configurations, a value of 1Bh should be used.

Look-Up Table Registers

Look-Up Table Address Register REG[24h]							RW
LUT Address Bit 7	LUT Address Bit 6	LUT Address Bit 5	LUT Address Bit 4	LUT Address Bit 3	LUT Address Bit 2	LUT Address Bit 1	LUT Address Bit 0

bits 7–0 LUT Address Bits [7:0]

These 8 bits control a pointer into the Look-Up Tables (LUT). The S1D13505 has three 256-position, 4-bit wide LUTs, one for each of red, green, and blue – refer to “*Look-Up Table Architecture*” for details.

This register selects which LUT entry is read/write accessible through the LUT Data Register (REG[26h]). Writing the LUT Address Register automatically sets the pointer to the Red LUT. Accesses to the LUT Data Register automatically increment the pointer.

For example, writing a value 03h into the LUT Address Register sets the pointer to R[3]. A subsequent access to the LUT Data Register accesses R[3] and moves the pointer onto G[3]. Subsequent accesses to the LUT Data Register move the pointer onto B[3], R[4], G[4], B[4], R[5], etc. Note that the RGB data is inserted into the LUT after the Blue data is written, i.e. all three colors must be written before the LUT is updated.

Look-Up Table Data Register REG[26h]							RW
LUT Data Bit 3	LUT Data Bit 2	LUT Data Bit 1	LUT Data Bit 0	n/a	n/a	n/a	n/a

bits 7–4 LUT Data

This register is used to read/write the RGB Look-Up Tables. This register accesses the entry at the pointer controlled by the Look-Up Table Address Register (REG[24h]) – see above.

Accesses to the Look-Up Table Data Register automatically increment the pointer. Note that the RGB data is inserted into the LUT after the Blue data is written, i.e. all three colors must be written before the LUT is updated.

Ink/Cursor Registers

Ink/Cursor Control Register REG[27h]							RW
Ink/Cursor Mode Bit 1	Ink/Cursor Mode Bit 0	n/a	n/a	Cursor High Threshold Bit 3	Cursor High Threshold Bit 2	Cursor High Threshold Bit 1	Cursor High Threshold Bit 0

bits 7–6 Ink/Cursor Control Bits [1:0]

These bits select the operating mode of the Ink/Cursor circuitry. See table below.

Table 8-17 Ink/Cursor Selection

REG[27h]		Operating Mode
Bit 7	Bit 6	
0	0	inactive
0	1	Cursor
1	0	Ink
1	1	reserved

bits 3–0 Ink/Cursor FIFO Threshold Bits [3:0]

These bits specify the Ink/Cursor FIFO depth required to sustain uninterrupted display fetches. When these bits are all 0, the Ink/Cursor FIFO depth is calculated automatically.

Cursor X Position Register 0 REG[28h]								RW
Cursor X Position Bit 7	Cursor X Position Bit 6	Cursor X Position Bit 5	Cursor X Position Bit 4	Cursor X Position Bit 3	Cursor X Position Bit 2	Cursor X Position Bit 1	Cursor X Position Bit 0	

Cursor X Position Register 1 REG[29h]								RW
Reserved	n/a	n/a	n/a	n/a	n/a	Cursor X Position Bit 9	Cursor X Position Bit 8	

REG[2Bh] bit 7**Reserved**

This bit must be set to 0.

REG[2Ah] bits 7–0**Cursor Y Position Bits [9:0]****REG[2Bh] bits 1–0**

In Cursor mode, this 10-bit register is used to program the vertical pixel position of the Cursor's top left pixel.

This register must be set to 0 in Ink mode.

Note: The Cursor X Position register must be set during VNDP (vertical non-display period). Check the VNDP status bit (REG[0Ah] bit 7) to determine if you are in VNDP, then update the register.

Cursor Y Position Register 0

REG[2Ah]							RW
Cursor Y Position Bit 7	Cursor Y Position Bit 6	Cursor Y Position Bit 5	Cursor Y Position Bit 4	Cursor Y Position Bit 3	Cursor Y Position Bit 2	Cursor Y Position Bit 1	Cursor Y Position Bit 0

Cursor Y Position Register 1

REG[2Bh]							RW
Reserved	n/a	n/a	n/a	n/a	n/a	Cursor Y Position Bit 9	Cursor Y Position Bit 8

REG[2Bh] bit 7 **Reserved**
This bit must be set to 0.

REG[2Ah] bits 7–0 **Cursor Y Position Bits [9:0]**

REG[2Bh] bits 1–0 In Cursor mode, this 10-bit register is used to program the vertical pixel position of the Cursor's top left pixel.
This register must be set to 0 in Ink mode.

Note: The Cursor Y Position register must be set during VNDP (vertical non-display period). Check the VNDP status bit (REG[0Ah] bit 7) to determine if you are in VNDP, then update the register.

Ink/Cursor Color 0 Register 0

REG[2Ch]							RW
Cursor Color 0 Bit 7	Cursor Color 0 Bit 6	Cursor Color 0 Bit 5	Cursor Color 0 Bit 4	Cursor Color 0 Bit 3	Cursor Color 0 Bit 2	Cursor Color 0 Bit 1	Cursor Color 0 Bit 0

Ink/Cursor Color 0 Register 1

REG[2Dh]							RW
Cursor Color 0 Bit 15	Cursor Color 0 Bit 14	Cursor Color 0 Bit 13	Cursor Color 0 Bit 12	Cursor Color 0 Bit 11	Cursor Color 0 Bit 10	Cursor Color 0 Bit 9	Cursor Color 0 Bit 8

REG[2Ch] bits 7–0 **Ink/Cursor Color 0 Bits [15:0]**

REG[2Dh] bits 7–0 **These bits define the 5-6-5 RGB Ink/Cursor color 0.**

Ink/Cursor Color 1 Register 0

REG[2Eh]							RW
Cursor Color 1 Bit 7	Cursor Color 1 Bit 6	Cursor Color 1 Bit 5	Cursor Color 1 Bit 4	Cursor Color 1 Bit 3	Cursor Color 1 Bit 2	Cursor Color 1 Bit 1	Cursor Color 1 Bit 0

Ink/Cursor Color 1 Register 1

REG[2Fh]							RW
Cursor Color 1 Bit 15	Cursor Color 1 Bit 14	Cursor Color 1 Bit 13	Cursor Color 1 Bit 12	Cursor Color 1 Bit 11	Cursor Color 1 Bit 10	Cursor Color 1 Bit 9	Cursor Color 1 Bit 8

REG[2Eh] bits 7–0 **Ink/Cursor Color 1 Bits [15:0]**

REG[2Fh] bits 7–0 **These bits define the 5-6-5 RGB Ink/Cursor color 1.**

Ink/Cursor Start Address Select Register

REG[30h]

RW

Ink/Cursor Start Address Select Bit 7	Ink/Cursor Start Address Select Bit 6	Ink/Cursor Start Address Select Bit 5	Ink/Cursor Start Address Select Bit 4	Ink/Cursor Start Address Select Bit 3	Ink/Cursor Start Address Select Bit 2	Ink/Cursor Start Address Select Bit 1	Ink/Cursor Start Address Select Bit 0
--	--	--	--	--	--	--	--

bits 7–0 Ink/Cursor Start Address Select Bits [7:0]

These bits define the start address for the Ink/Cursor buffer. The Ink/Cursor buffer must be positioned where it does not conflict with the image buffer and half-frame buffer – see Memory Mapping for details.

The start address for the Ink/Cursor buffer is programmed as shown in the following table where Display Buffer Size represents the size in bytes of the attached DRAM device (see MD[7:6] in “*Summary of Configuration Options*”):

Table 8-18 Ink/Cursor Start Address Encoding

Ink/Cursor Start Address Bits [7:0]	Start Address (Bytes)
0	Display Buffer Size - 1024
n = 255...1	Display Buffer Size - (n × 8192)

The Ink/Cursor image is stored contiguously. The address offset from the starting word of line n to the starting word of line n+1 is calculated as follows:

Ink Address Offset (words) = REG[04h] + 1

Cursor Address Offset (words) = 8

Alternate FRM Register

REG[31h]

RW

Alternate FRM Bit 7	Alternate FRM Bit 6	Alternate FRM Bit 5	Alternate FRM Bit 4	Alternate FRM Bit 3	Alternate FRM Bit 2	Alternate FRM Bit 1	Alternate FRM Bit 0
------------------------	------------------------	------------------------	------------------------	------------------------	------------------------	------------------------	------------------------

bits 7–0 Alternate Frame Rate Modulation Select

Register that controls the alternate FRM scheme. When all bits are set to zero, the default FRM is selected. For single passive, or dual passive with the half frame buffer enabled, either the original or the alternate FRM scheme may be used. The alternate FRM scheme may produce more visually appealing output. The following table shows the recommended alternate FRM scheme values.

Table 8-19 Recommended Alternate FRM Scheme

Panel Mode	Register Value
Single Passive	0000 0000 or 1111 1111
Dual Passive w/Half Frame Buffer Enabled	0000 0000 or 1111 1010
Dual Passive w/Half Frame Buffer Disabled	1111 1111

9 DISPLAY BUFFER

The system addresses the display buffer through the CS#, M/R#, and AB[20:0] input pins. When CS# = 0 and M/R# = 1, the display buffer is addressed by bits AB[20:0]. See the table below:

Table 9-1 S1D13505 Addressing

CS#	M/R#	Access
0	0	Register access: <ul style="list-style-type: none"> • REG[00h] is addressed when AB[5:0] = 0 • REG[01h] is addressed when AB[5:0] = 1 • REG[n] is addressed when AB[5:0] = n
0	1	Memory access: the 2M byte display buffer is addressed by AB[20:0]
1	×	S1D13505 not selected

The display buffer address space is always 2M bytes. However, the physical display buffer may be either 512K bytes or 2M bytes – see “*Summary of Configuration Options*”.

The display buffer can contain an image buffer, one or more Ink/Cursor buffers, and a half-frame buffer.

A 512K byte display buffer is replicated in the 2M byte address space – see the figure below.

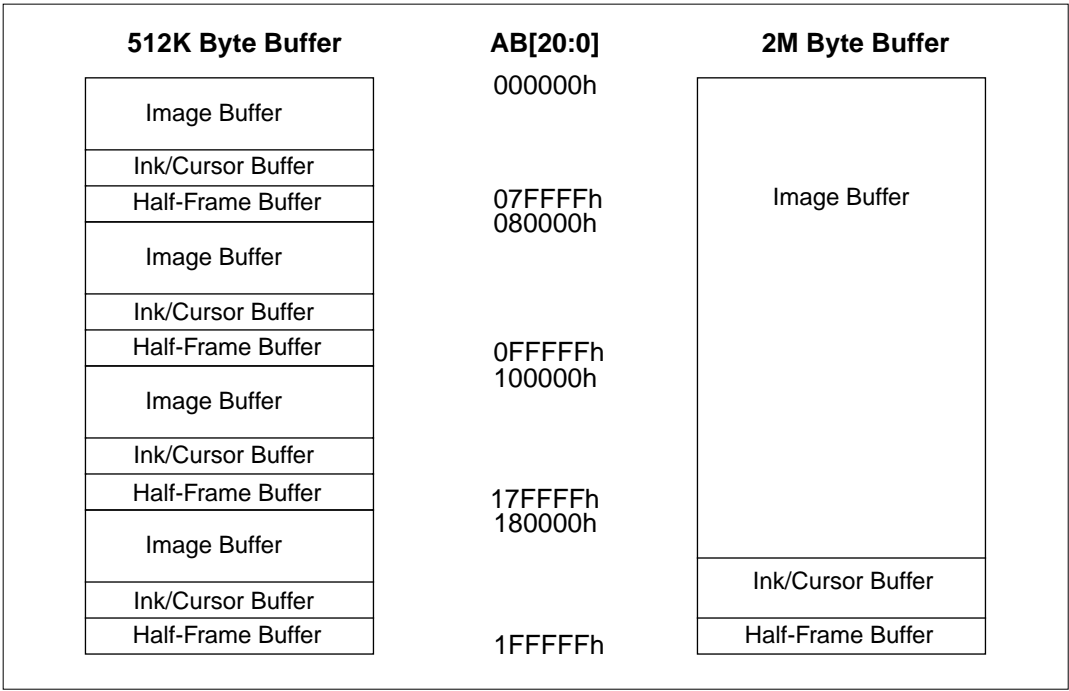


Figure 9-1 Display Buffer Addressing

10 DISPLAY CONFIGURATION

10.1 Display Mode Data Format

The following diagrams show the display mode data formats for a little-endian system.

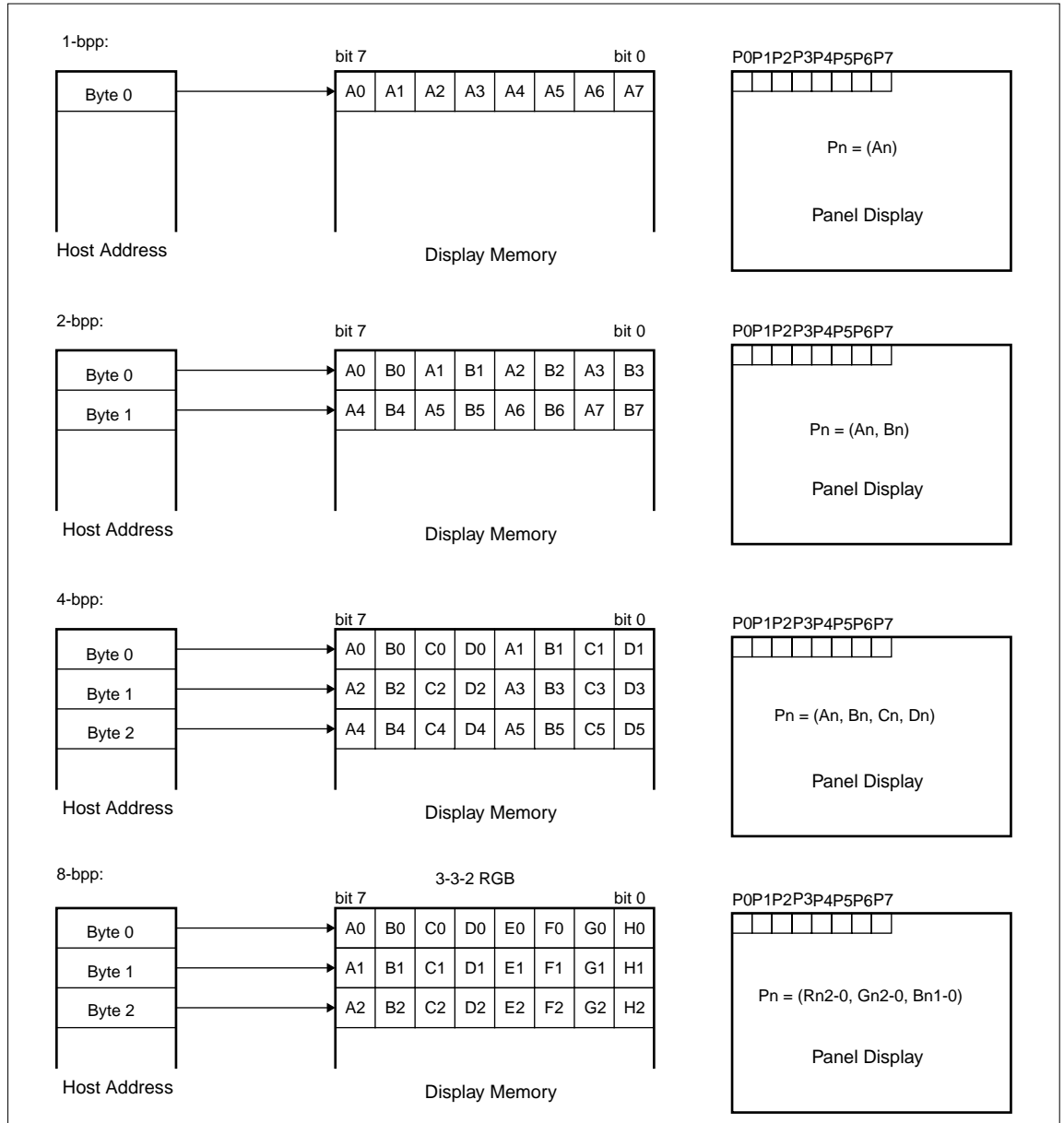


Figure 10-1 1/2/4/8 Bit-Per-Pixel Format Memory Organization

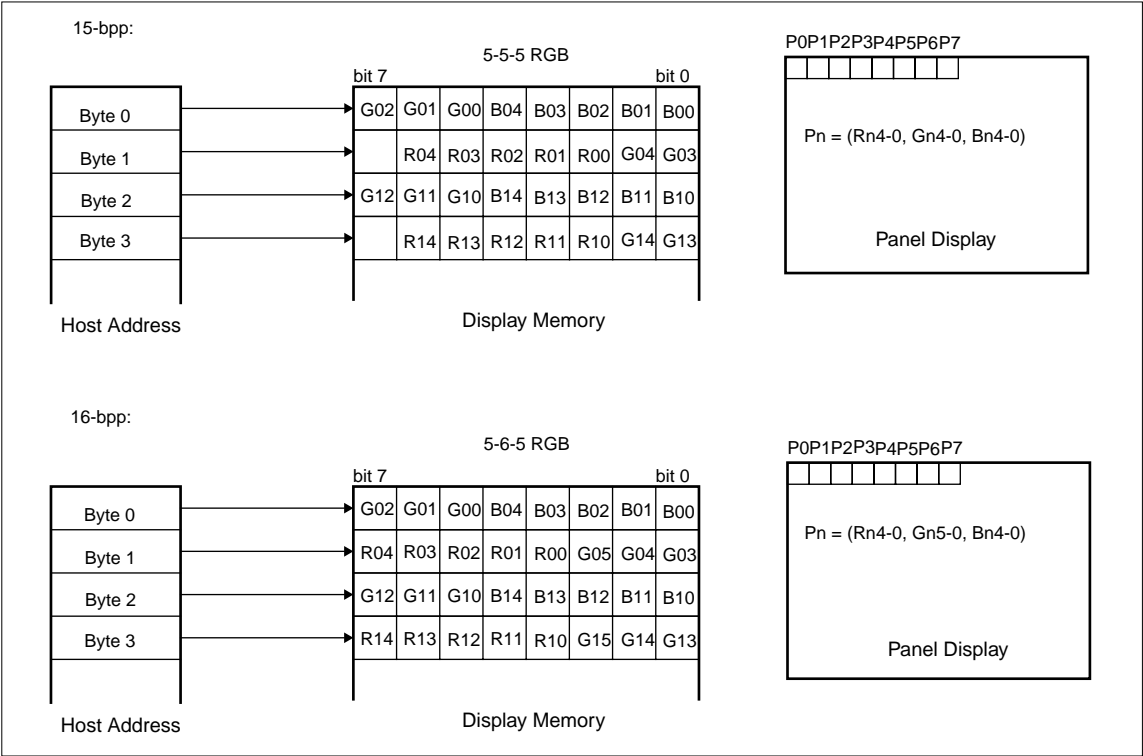


Figure 10-2 15/16 Bit-Per-Pixel Format Memory Organization

- Notes:** 1. The Host-to-Display mapping shown here is for a little-endian system.
2. For 15/16 bpp formats, Rn, Gn, Bn represent the red, green, and blue color components.

10.2 Image Manipulation

The figure below shows how Screen 1 and 2 images are stored in the image buffer and positioned on the display. Screen 1 and Screen 2 can be parts of a larger virtual image or images.

- (REG[17h],REG[16h]) defines the width of the virtual image(s).
- (REG[12h],REG[11h],REG[10h]) defines the starting word of the Screen 1, (REG[15h],REG[14h],REG[13h]) defines the starting word of the Screen 2.
- REG[18h] bits [3:0] define the starting pixel within the starting word for Screen 1, REG[18h] bits [7:4] define the starting pixel within the starting word for Screen 2.
- (REG[0Fh],REG[0Eh]) define the last line of Screen 1, the remainder of the display is taken up by Screen 2.

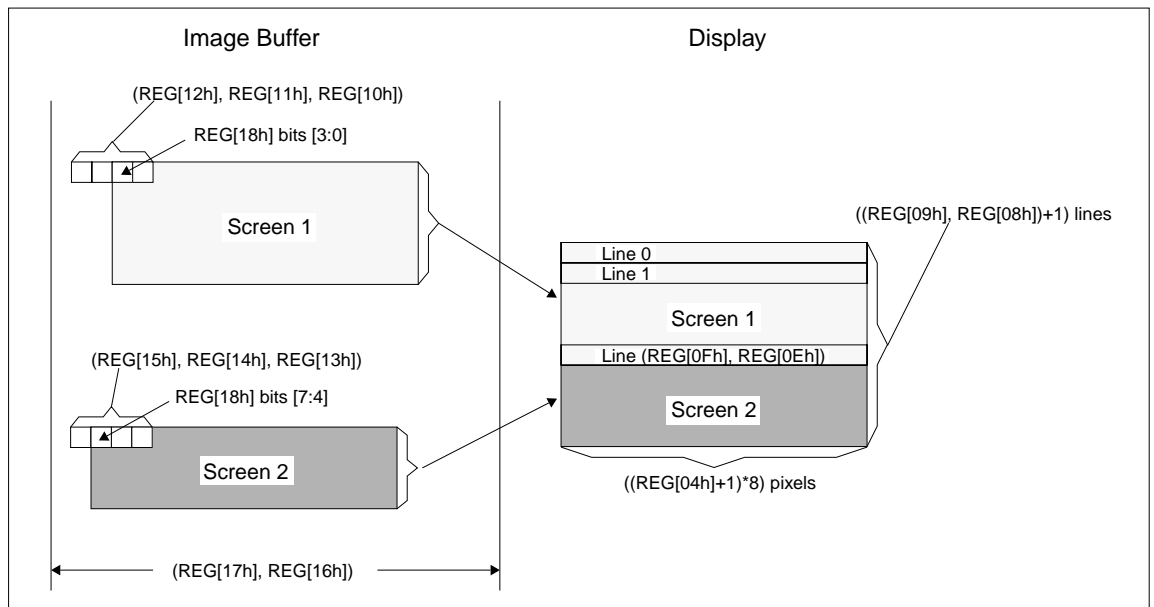


Figure 10-3 Image Manipulation

11 LOOK-UP TABLE ARCHITECTURE

The following figures are intended to show the display data output path only.

11.1 Monochrome Modes

The green Look-Up Table (LUT) is used for all monochrome modes.

1 Bit-Per-Pixel Monochrome Mode

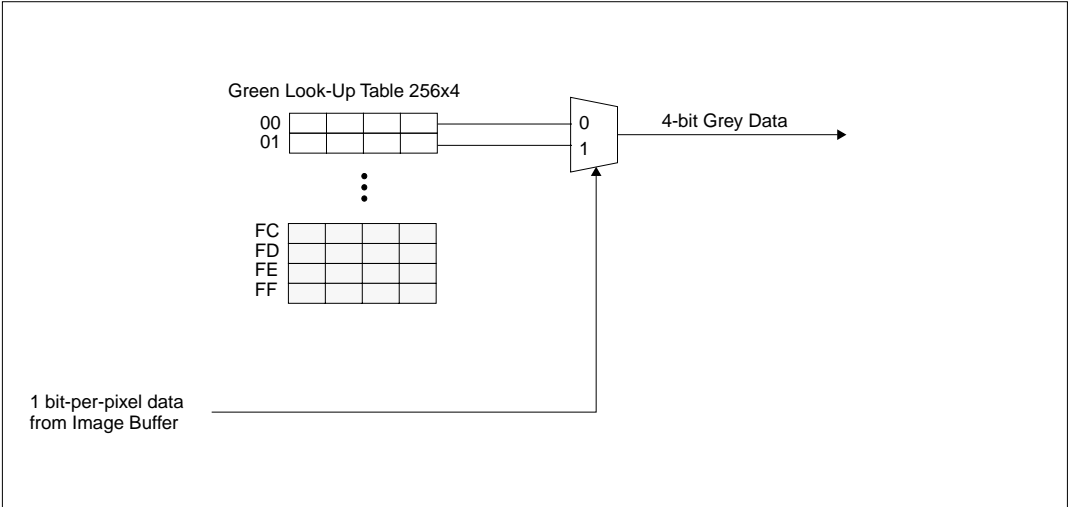


Figure 11-1 1 Bit-per-pixel Monochrome Mode Data Output Path

2 Bit-Per-Pixel Monochrome Mode

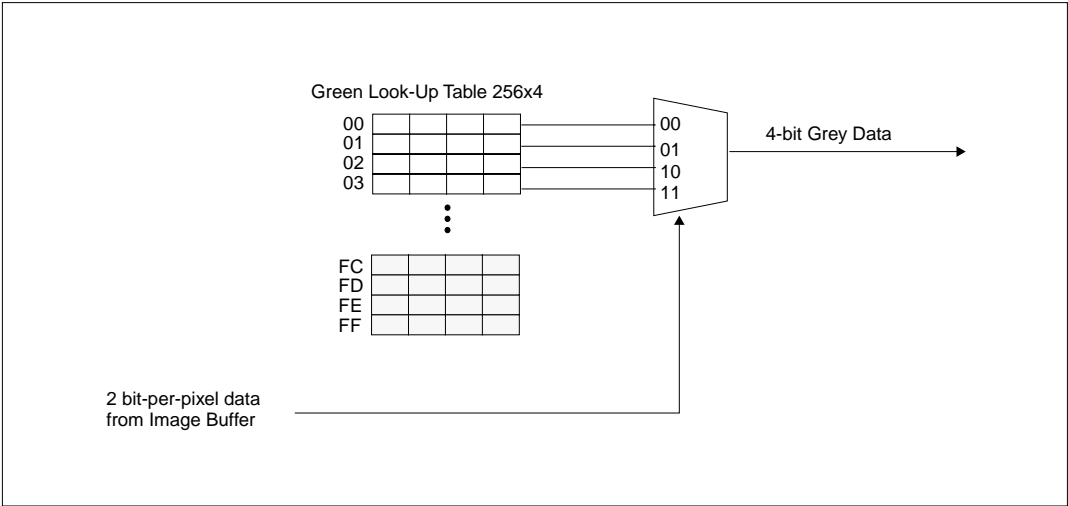


Figure 11-2 2 Bit-per-pixel Monochrome Mode Data Output Path

4 Bit-Per-Pixel Monochrome Mode

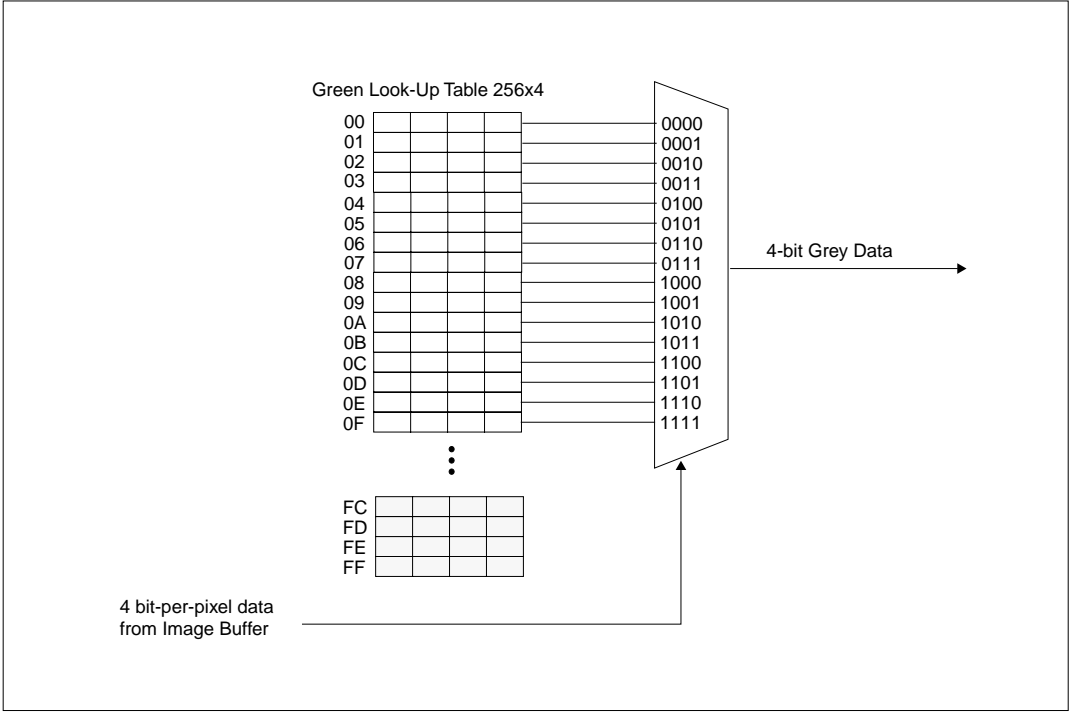


Figure 11-3 4 Bit-per-pixel Monochrome Mode Data Output Path

11.2 Color Display Modes

1 Bit-Per-Pixel Color Mode

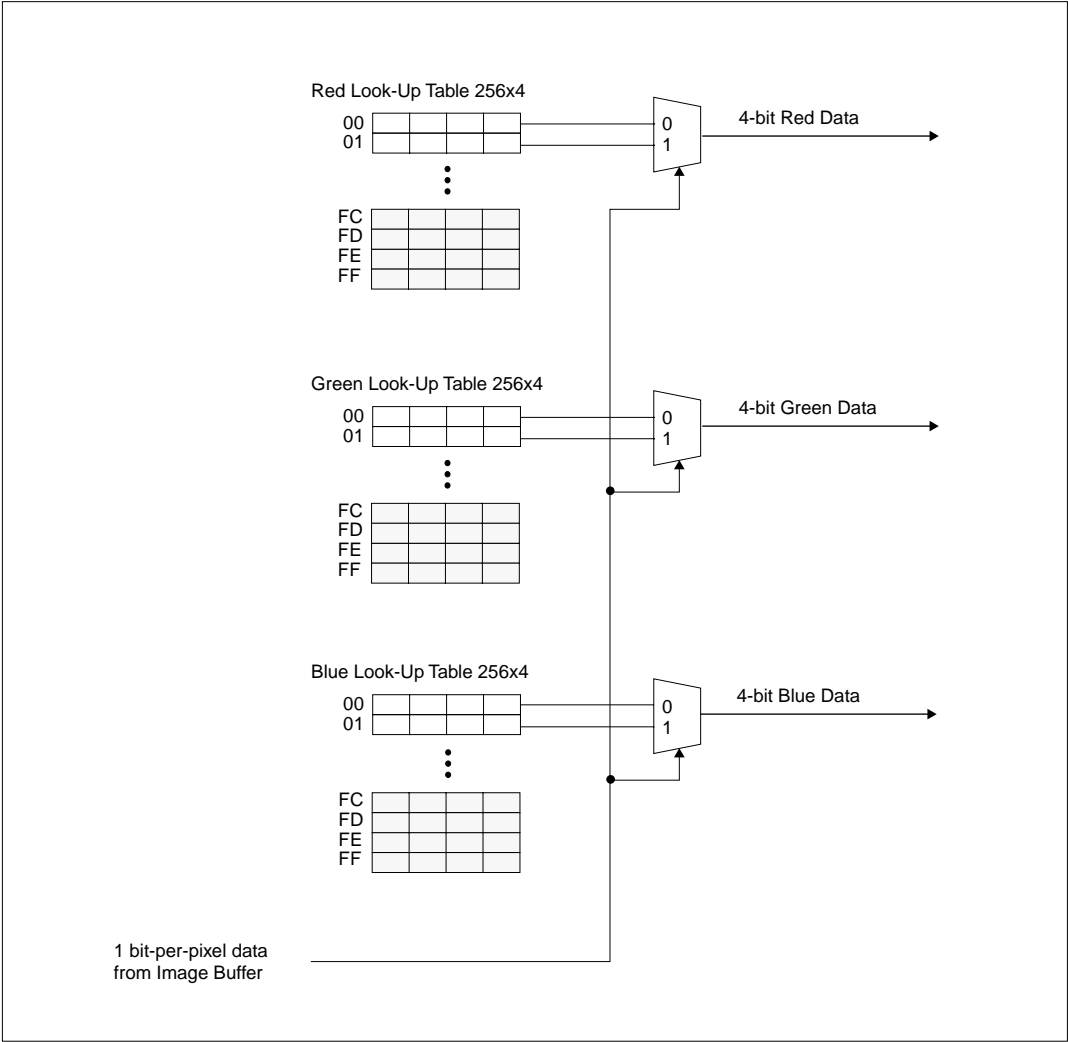


Figure 11-4 1 Bit-per-pixel Color Mode Data Output Path

2 Bit-Per-Pixel Color Mode

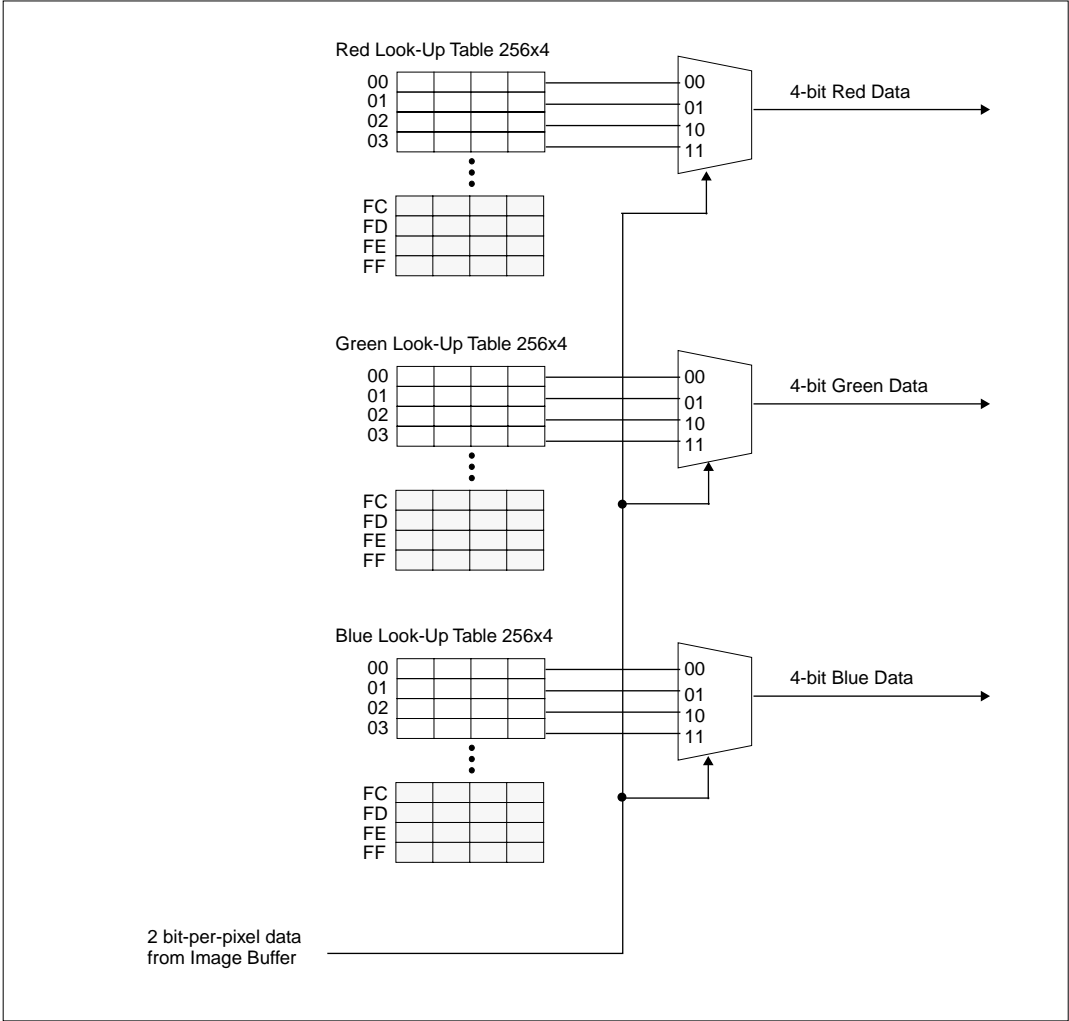


Figure 11-5 2 Bit-per-pixel Color Mode Data Output Path

4 Bit-Per-Pixel Color Mode

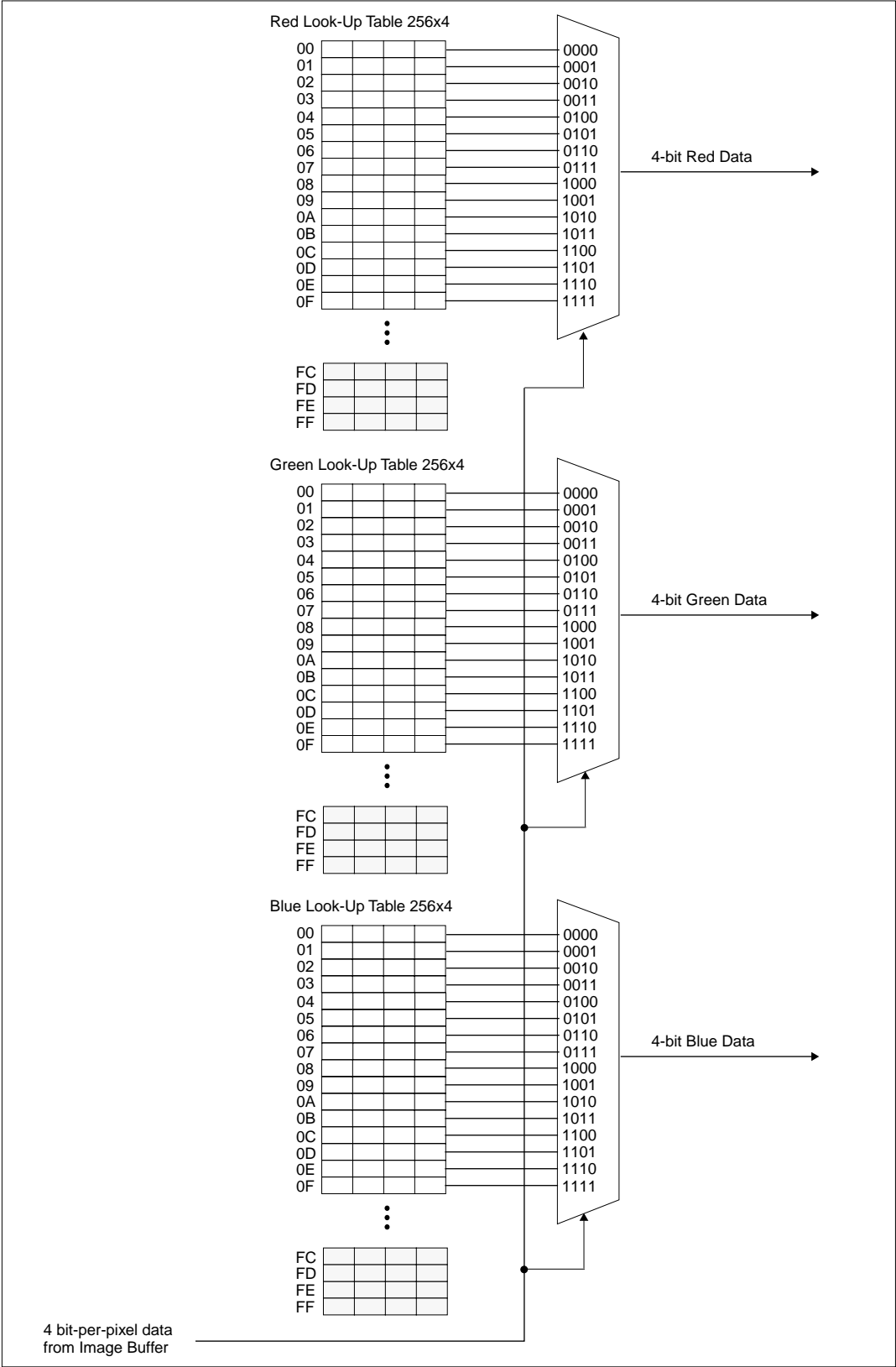


Figure 11-6 4 Bit-per-pixel Color Mode Data Output Path

8 Bit-Per-Pixel Color Mode

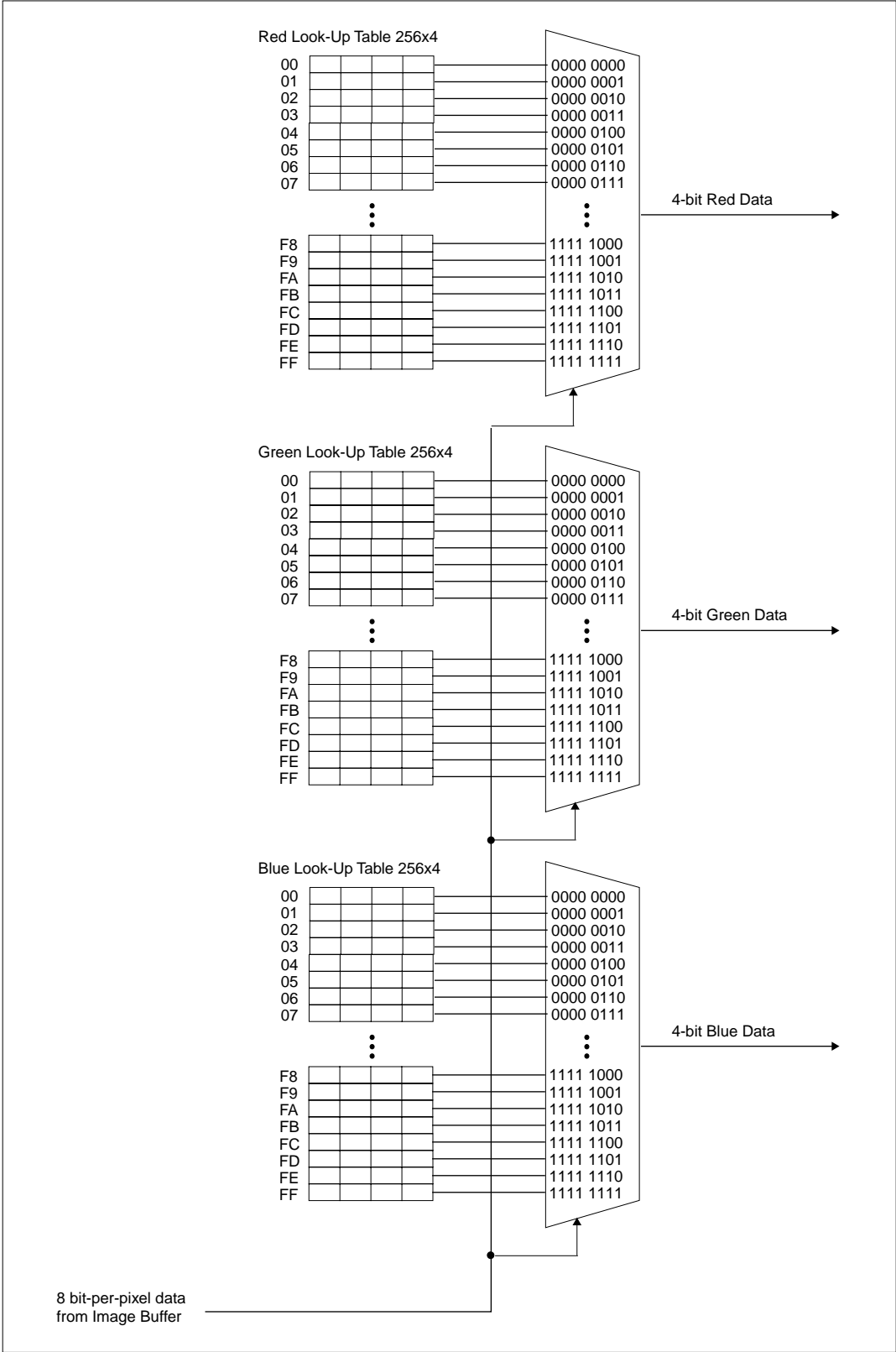


Figure 11-7 8 Bit-per-pixel Color Mode Data Output Path

15/16 Bit-Per-Pixel Color Modes

The LUT is bypassed and the color data is directly mapped for this color mode – See “Display Configuration” on page 1-103.

12 INK/CURSOR ARCHITECTURE

12.1 Ink/Cursor Buffers

The Ink/Cursor buffers contain formatted image data for the Ink Layer or Hardware Cursor. There may be several Ink/Cursor images stored in the display buffer but only one may be active at any given time.

The active Ink/Cursor buffer is selected by the Ink/Cursor Start Address register (REG[30h]). This register defines the start address for the active Ink/Cursor buffer. The Ink/Cursor buffer must be positioned where it does not conflict with the image buffer and half-frame buffer. The start address for the Ink/Cursor buffer is programmed as shown in the following table:

Table 12-1 Ink/Cursor Data Format

Ink/Cursor Start Address Bits [7:0]	Start Address (Bytes)	Comments
0	Display Buffer Size - 1024	This default value is suitable for a cursor when there is no half-frame buffer.
$n = 255 \dots 1$	Display Buffer Size - ($n \times 8192$)	These positions can be used to: <ul style="list-style-type: none"> • position an Ink buffer at the top of the display buffer; • position an Ink buffer between the image and half-frame buffers; • position a Cursor buffer between the image and half-frame buffers; • select from a multiple of Cursor buffers.

The Ink/Cursor image is stored contiguously. The address offset from the starting word of line n to the starting word of line $n+1$ is calculated as follows:

$$\begin{aligned} \text{Ink Address Offset (words)} &= \text{REG}[04h] + 1 \\ \text{Cursor Address Offset (words)} &= 8 \end{aligned}$$

12.2 Ink/Cursor Data Format

The Ink/Cursor image is always 2 bit-per-pixel. The following diagram shows the Ink/Cursor data format for a little-endian system.

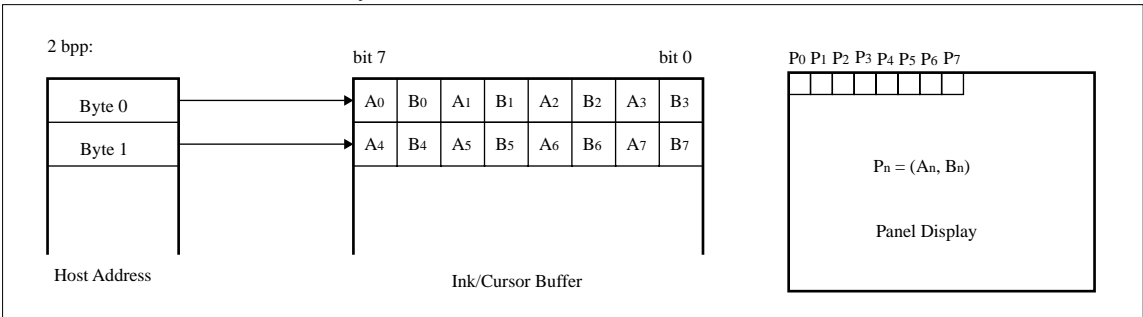


Figure 12-1 Ink/Cursor Data Format

The image data for pixel n , (A_n, B_n) , selects the color for pixel n as follows:

Table 12-2 Ink/Cursor Color Select

(A_n, B_n)	Color	Comments
00	Color 0	Ink/Cursor Color 0 Register, (REG[2Dh], REG[2Ch])
01	Color 1	Ink/Cursor Color 1 Register, (REG[2Fh], REG[2Eh])
10	Background	Ink/Cursor is transparent – show background
11	Inverted Background	Ink/Cursor is transparent – show inverted background

12.3 Ink/Cursor Image Manipulation

Ink Image

The Ink image should always start at the top left pixel, i.e. Cursor X Position and Cursor Y Position registers should always be set to zero. The width and height of the ink image are automatically calculated to completely cover the display.

Cursor Image

The Cursor image size is always 64x64 pixels. The Cursor X Position and Cursor Y Position registers specify the position of the top left pixel. The following diagram shows how to position a cursor.

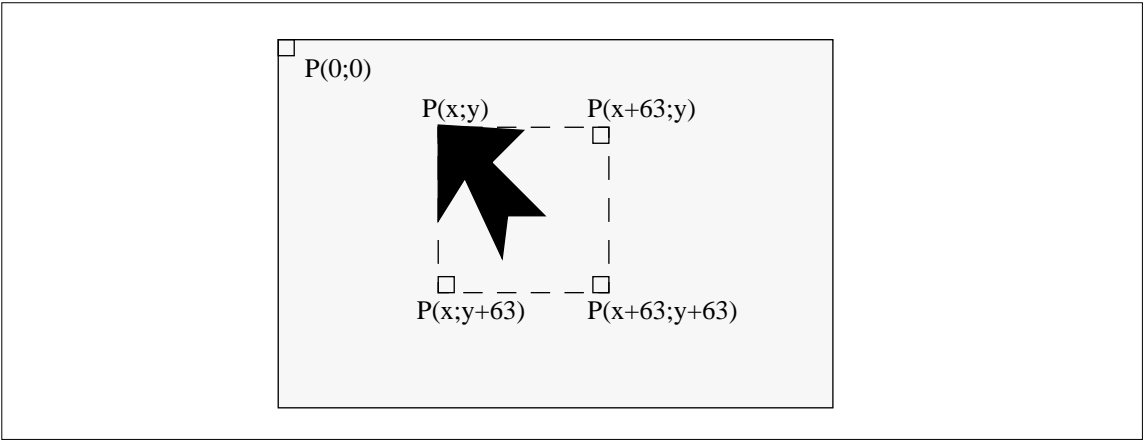


Figure 12-2 Cursor Positioning

where $x = (\text{REG}[29\text{h}] \text{ bits } [1:0], \text{REG}[28\text{h}])$ $\text{REG}[29\text{h}] \text{ bit } 7 = 0$
 $y = (\text{REG}[2\text{Bh}] \text{ bits } [1:0], \text{REG}[2\text{Ah}])$ $\text{REG}[2\text{Bh}] \text{ bit } 7 = 0$

Note: There is no means to set a negative cursor position. If a cursor must be set to a negative position, this must be dealt with through software.

13 SWIVELVIEW™

13.1 Concept

Computer displays are refreshed in landscape – from left to right and top to bottom; computer images are stored in the same manner. When a display is used in SwivelView™ it becomes necessary to rotate the display buffer image by 90°. The S1D13505 supports SwivelView™ by rotating the image 90° clockwise as it is written to the display buffer. The rotation is done in hardware and is transparent to the programmer for all display buffer reads and writes.

SwivelView™ uses a 1024 × 1024 pixel virtual image. The following figures show how the programmer sees the image and how the image is actually stored in the display buffer. The display is refreshed in the following sense: C–A–D–B. The application image is written to the S1D13505 in the following sense: A–B–C–D. The S1D13505 rotates and stores the application image in the following sense: C–A–D–B, the same sense as display refresh.

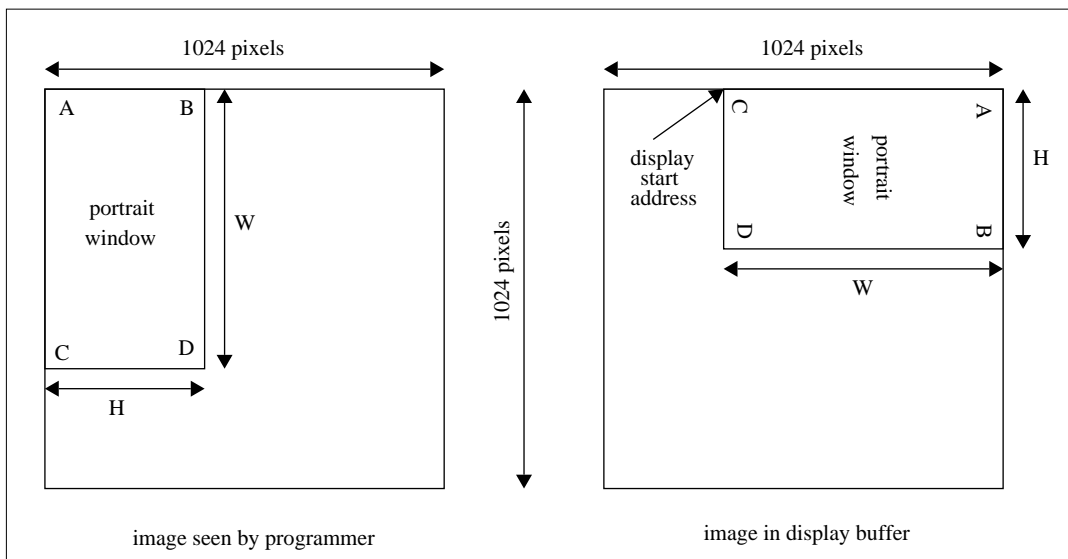


Figure 13-1 Relationship Between the Screen Image and the Image Residing in the Display Buffer

Note: The image must be written with a 1024 pixel offset between adjacent lines (e.g. 1024 bytes for 8 bpp mode or 2048 bytes for 16 bpp mode) and a display start address that is non-zero.

13.2 Image Manipulation in SwivelView™

Display Start Address

It can be seen from Figure 13-1 that the top left pixel of the display is not at the top left corner of the virtual image, i.e. it is non-zero. The Display Start Address register must be set accordingly:

$$\begin{aligned} \text{Display Start Address (words)} &= (1024 - W) && \text{for 16 bpp mode} \\ &= (1024 - W) / 2 && \text{for 8 bpp mode} \end{aligned}$$

Memory Address Offset

The Memory Address Offset register must be set for a 1024 pixel offset:

$$\begin{aligned} \text{Memory Address Offset (words)} &= 1024 && \text{for 16 bpp mode} \\ &= 512 && \text{for 8 bpp mode} \end{aligned}$$

Horizontal Panning

Horizontal panning is achieved by changing the start address. Panning of the portrait window to the right by 1 pixel is achieved by adding 1024 pixels to the Display Start Address register (or subtracting if panning to the left).

- Panning to right by 1 pixel: add current start address by 1024 (16 bpp mode) or 512 (8 bpp mode).
- Panning to left by 1 pixel: subtract current start address by 1024 (16 bpp mode) or 512 (8 bpp mode).

How far the portrait window can be panned to the right is limited not only by 1024 pixels but also by the amount of physical memory installed.

Vertical Scrolling

Vertical scrolling is achieved by changing the Display Start Address register and/or changing the Pixel Panning register.

- Increment/decrement Display Start Address register in 8 bpp mode: scroll down/up by 2 lines.
- Increment/decrement Display Start Address register in 16 bpp mode: scroll down/up by 1 line.
- Increment/decrement Pixel Panning register in 8 bpp or 16 bpp mode: scroll down/up by 1 line.

13.3 Physical Memory Requirement

Because the programmer must now deal with a virtual display, the amount of image buffer required for a particular display mode has increased. The minimum amount of image buffer required is:

$$\begin{aligned} &\text{Minimum Required Image Buffer (bytes)} \\ &= (1024 \times H) \times 2 \quad \text{for 16 bpp mode} \\ &= (1024 \times H) \quad \text{for 8 bpp mode} \end{aligned}$$

For single panel, the required display buffer size is the same as the image buffer required. For dual panel, the display buffer required is the sum of the image buffer required and the half-frame buffer memory required. The half-frame buffer memory requirement is:

$$\begin{aligned} &\text{Half-Frame Buffer Memory (bytes)} \\ &= (W \times H) / 4 \quad \text{for color mode} \\ &= (W \times H) / 16 \quad \text{for monochrome mode} \end{aligned}$$

The half-frame buffer memory is always located at the top of the physical memory.

For simplicity the hardware cursor and ink layer memory requirement is ignored. The hardware cursor and ink layer memory must be located at 16K byte boundaries and it must not overlap the image buffer and half-frame buffer memory areas.

Even though the virtual display is 1024×1024 pixels, the actual panel window is always smaller. Thus it is possible for the display buffer size to be smaller than the virtual display but large enough to fit both the required image buffer and the half-frame buffer memory. This poses a maximum “accessible” horizontal virtual size limit.

$$\begin{aligned} &\text{Maximum Accessible Horizontal Virtual Size (pixels)} \\ &= (\text{Physical Memory} - \text{Half-Frame Buffer Memory}) / 2048 \quad \text{for 16 bpp mode} \\ &= (\text{Physical Memory} - \text{Half-Frame Buffer Memory}) / 1024 \quad \text{for 8 bpp mode} \end{aligned}$$

For example, a 640×480 single panel running 8 bpp mode requires 480K byte of image buffer and 0K byte of half-frame buffer memory. The virtual display size is 1024×1024 = 1M byte. The programmer may use a 512K byte DRAM which is smaller than the 1M byte virtual display but greater than the 480K byte minimum required image buffer. The maximum accessible horizontal virtual size is = (512K byte - 0K byte) / 1024 = 512. The programmer therefore has room to pan the portrait window to the right by 512 - 480 = 32 pixels. The programmer also should not read/write to the memory beyond the maximum accessible horizontal virtual size because that memory is either reserved for the half-frame buffer or not associated with any real memory at all.

The following table summarizes the DRAM size requirement for SwivelView™ using different panel sizes and display modes. Note that DRAM size for the S1D13505 is limited to either 512K byte or 2M byte. The calculation is based on the minimum required image buffer size. The calculated minimum display buffer size is based on the image buffer and the half-frame buffer only; it does not take into account the hardware cursor/ink layer and so it may or may not be sufficient to support it – this is noted in the table. The hardware cursor requires 1K byte of memory and the 2-bit ink layer requires (W × H) / 4 bytes of memory; both must reside at 16K byte boundaries but only one is supported at a time. The table shows only one possible sprite/ink layer location – at the highest possible 16K byte boundary below the half-frame buffer which is always at the top.

Table 13-1 Minimum DRAM Size Required for SwivelView™

Panel Size	Panel Type		Display Mode	Display Buffer Size	Half-Frame Buffer Size	Minimum DRAM Size	Sprite/Ink Layer Buffer Size	Ink/Cursor Layer Location				
320 × 240	Single	Color	8 bpp	240KB	0KB	512KB	1KB/18.75KB	496KB/480KB				
			16 bpp	480KB								
		Mono	8 bpp	240KB								
			16 bpp	480KB								
	Dual	Color	8 bpp	240KB	18.75KB				512KB	1KB/18.75KB	480KB/464KB	
			16 bpp	480KB							480KB/--	
		Mono	8 bpp	240KB	4.69KB						496KB/480KB	
			16 bpp	480KB								
640 × 480	Single	Color	8 bpp	480KB	0KB	2MB	1KB/75KB	496KB/--				
			16 bpp	960KB				2032KB/1968KB				
			Mono	8 bpp				480KB			512KB	496KB/--
				16 bpp				960KB			2MB	2032K/1968K
		Dual	Color	8 bpp	480KB	75KB		512KB	1KB/75KB	496KB/--		
				16 bpp	960KB					2032KB/1968KB		
			Mono	8 bpp	480KB	18.75KB				512KB	496KB/--	
				16 bpp	960KB					2MB	2032KB/1968KB	
800 × 600	Single	Color	8 bpp	600KB	0KB	2MB	1KB/117.19KB	2032KB/1920KB				
			16 bpp	1.2MB								
			Mono	8 bpp						600KB		
				16 bpp						1.2MB		
		Dual	Color	8 bpp	600KB	117.19KB			2MB	1KB/117.19KB	2032KB/1920KB	
				16 bpp	1.2MB							
			Mono	8 bpp	600KB	29.30KB						
				16 bpp	1.2MB							

Where KB = K bytes and MB = 1024K bytes

13.4 Limitations

The following limitations apply to SwivelView™:

- Only 8 bpp and 16 bpp modes are supported – 1/2/4 bpp modes are not supported.
- Hardware cursor and ink layer images are not rotated – software rotation must be used. SwivelView™ must be turned off when the programmer is accessing the sprite or the ink layer.
- Split screen images appear side-by-side, i.e. the portrait display is split vertically.
- Pixel panning works vertically.

14 CLOCKING

14.1 Maximum MCLK: PCLK Ratios

Table 14-1 Maximum PCLK Frequency with EDO-DRAM

Ink	Display Type	NRC	Maximum PCLK Allowed				
			1 bpp	2 bpp	4 bpp	8 bpp	16 bpp
off	<ul style="list-style-type: none">• Single Panel.• CRT.• Dual Monochrome/Color Panel with Half Frame Buffer Disabled.• Simultaneous CRT + Single Panel.• Simultaneous CRT + Dual Monochrome/Color Panel with Half Frame Buffer Disabled.	5, 4, 3	MCLK				
	<ul style="list-style-type: none">• Dual Monochrome Panel with Half Frame Buffer Enabled.• Simultaneous CRT + Dual Monochrome Panel with Half Frame Buffer Enable.	5	MCLK/2	MCLK/2	MCLK/2	MCLK/2	MCLK/3
		4	MCLK/2	MCLK/2	MCLK/2	MCLK/2	MCLK/3
	3	MCLK	MCLK	MCLK/2	MCLK/2	MCLK/2	
	<ul style="list-style-type: none">• Dual Color Panel with Half Frame Buffer Enabled.• Simultaneous CRT + Dual Color Panel with Half Frame Buffer Enable.	5	MCLK/2	MCLK/2	MCLK/2	MCLK/3	MCLK/3
		4	MCLK/2	MCLK/2	MCLK/2	MCLK/2	MCLK/3
		3	MCLK/2	MCLK/2	MCLK/2	MCLK/2	MCLK/3
	on	<ul style="list-style-type: none">• Single Panel.• CRT.• Dual Monochrome/Color Panel with Half Frame Buffer Disabled.• Simultaneous CRT + Single Panel.• Simultaneous CRT + Dual Monochrome/Color Panel with Half Frame Buffer Disabled.	5	MCLK/2	MCLK/2	MCLK/2	MCLK/2
4			MCLK	MCLK	MCLK/2	MCLK/2	MCLK/2
3		MCLK	MCLK	MCLK	MCLK/2	MCLK/2	
<ul style="list-style-type: none">• Dual Monochrome Panel with Half Frame Buffer Enabled.• Simultaneous CRT + Dual Monochrome Panel with Half Frame Buffer Enable.		5	MCLK/2	MCLK/3	MCLK/3	MCLK/3	MCLK/3
		4	MCLK/2	MCLK/2	MCLK/2	MCLK/3	MCLK/3
		3	MCLK/2	MCLK/2	MCLK/2	MCLK/2	MCLK/3
<ul style="list-style-type: none">• Dual Color Panel with Half Frame Buffer Enabled.• Simultaneous CRT + Dual Color Panel with Half Frame Buffer Enable.		5	MCLK/3	MCLK/3	MCLK/3	MCLK/3	MCLK/4
		4	MCLK/2	MCLK/2	MCLK/3	MCLK/3	MCLK/3
		3	MCLK/2	MCLK/2	MCLK/2	MCLK/2	MCLK/3

Table 14-2 Maximum PCLK Frequency with FPM-DRAM

Ink	Display Type	Nrc	Maximum PCLK Allowed				
			1 bpp	2 bpp	4 bpp	8 bpp	16 bpp
off	<ul style="list-style-type: none"> Single Panel. CRT. Dual Monochrome/Color Panel with Half Frame Buffer Disabled. Simultaneous CRT + Single Panel. Simultaneous CRT + Dual Monochrome/Color Panel with Half Frame Buffer Disabled. 	5, 4, 3	MCLK				
	Dual Monochrome with Half Frame Buffer Enabled.	5	MCLK/2	MCLK/2	MCLK/2	MCLK/2	MCLK/3
	Simultaneous CRT + Dual Monochrome Panel with Half Frame Buffer Enable.	4	MCLK/2	MCLK/2	MCLK/2	MCLK/2	MCLK/2
		3	MCLK	MCLK	MCLK	MCLK/2	MCLK/2
	Dual Color with Half Frame Buffer Enabled.	5	MCLK/2	MCLK/2	MCLK/2	MCLK/2	MCLK/3
	Simultaneous CRT + Dual Color Panel with Half Frame Buffer Enable.	4	MCLK/2	MCLK/2	MCLK/2	MCLK/2	MCLK/3
		3	MCLK/2	MCLK/2	MCLK/2	MCLK/2	MCLK/2
	Single Panel.	5	MCLK/2	MCLK/2	MCLK/2	MCLK/2	MCLK/3
		4	MCLK	MCLK	MCLK/2	MCLK/2	MCLK/2
on	<ul style="list-style-type: none"> CRT. Dual Monochrome/Color Panel with Half Frame Buffer Disabled. Simultaneous CRT + Single Panel. Simultaneous CRT + Dual Monochrome/Color Panel with Half Frame Buffer Disabled. 	3	MCLK	MCLK	MCLK	MCLK/2	MCLK/2
	Dual Monochrome with Half Frame Buffer Enabled.	5	MCLK/2	MCLK/2	MCLK/3	MCLK/3	MCLK/3
	Simultaneous CRT + Dual Monochrome Panel with Half Frame Buffer Enable.	4	MCLK/2	MCLK/2	MCLK/2	MCLK/2	MCLK/3
		3	MCLK/2	MCLK/2	MCLK/2	MCLK/2	MCLK/3
	Dual Color with Half Frame Buffer Enabled.	5	MCLK/3	MCLK/3	MCLK/3	MCLK/3	MCLK/4
	Simultaneous CRT + Dual Color Panel with Half Frame Buffer Enable.	4	MCLK/2	MCLK/2	MCLK/2	MCLK/3	MCLK/3
		3	MCLK/2	MCLK/2	MCLK/2	MCLK/2	MCLK/3

14.2 Frame Rate Calculation

The frame rate is calculated using the following formula:

$$\text{FrameRate} = \frac{\text{PCLKmax}}{(\text{HDP} + \text{HNDP}) \times (\text{VDP} + \text{VNDP})}$$

Where:

VDP	= Vertical Display Period	= REG[09h] bits [1:0], REG[08h] bits [7:0] + 1
VNDP	= Vertical Non-Display Period	= REG[0Ah] bits [5:0] + 1
		= in table below
HDP	= Horizontal Display Period	= ((REG[04h] bits [6:0]) + 1) * 8Ts
HNDP	= Horizontal Non-Display Period	= ((REG[05h] bits [4:0]) + 1) * 8Ts
		= given in table below
Ts	= Pixel Clock	= PCLK

Table 14-3 Example Frame Rates with Ink Disabled

DRAM Type ¹ (Speed Grade)	Display	Resolution	Color Depth (bpp)	Maximum Pixel Clock (MHz)	Minimum Panel HNDP(T _s)	Maximum Frame Rate (Hz)	
						Panel ⁴	CRT
50ns EDO-DRAM MClk = 40MHz NRC = 4 NRP = 1.5 NRCD = 2	<ul style="list-style-type: none"> • Single Panel. • CRT. • Dual Monochrome/Color Panel with Half Frame Buffer Disabled.⁵ • Simultaneous CRT + Single Panel. • Simultaneous CRT + Dual Monochrome/Color Panel with Half Frame Buffer Disabled.⁵ 	800x600 ²	1/2/4/8	40	32	80	60
			16		56	78	60
		640x480	1/2/4/8		32	123	85
			16		56	119	85
		640x240	1/2/4/8		32	247	-
			16		56	242	-
		480x320	1/2/4/8		32	243	-
			16		56	232	-
		320x240	1/2/4/8		32	471	-
			16		56	441	-
	<ul style="list-style-type: none"> • Dual Color with Half Frame Buffer Enabled. • Dual Mono with Half Frame Buffer Enabled. 	800x600 ^{2,3}	1/2/4/8	20	32	80	-
			16	13.3	32	53	-
		640x480	1/2/4/8	20	32	123	-
			16	13.3	32	82	-
60ns EDO-DRAM MClk = 33MHz NRC = 4 NRP = 1.5 NRCD = 2	<ul style="list-style-type: none"> • Single Panel. • CRT. • Dual Mono/Color Panel with Half Frame Buffer Disabled.⁵ • Simultaneous CRT + Single Panel. • Simultaneous CRT + Dual Mono/Color Panel with Half Frame Buffer Disabled.⁵ 	800x600 ²	1/2/4/8	33	32	66	55
			16		56	65	55
		640x480	1/2/4/8		32	101	78
			16		56	98	78
		640x240	1/2/4/8		32	203	-
			16		56	200	-
		480x320	1/2/4/8		32	200	-
			16		56	196	-
		320x240	1/2/4/8		32	388	-
			16		56	380	-
	<ul style="list-style-type: none"> • Dual Color with Half Frame Buffer Enabled. • Dual Mono with Half Frame Buffer Enabled. 	800x600 ^{2,3}	1/2/4/8	16.5	32	66	-
			16	11	32	43	-
		640x480	1/2/4/8	16.5	32	103	-
			16	11	32	68	-
60ns FPM-DRAM MClk = 25MHz NRC = 4 NRP = 1.5 NRCD = 2	<ul style="list-style-type: none"> • Single Panel. • CRT. • Dual Mono/Color Panel with Half Frame Buffer Disabled.⁵ • Simultaneous CRT + Single Panel. • Simultaneous CRT + Dual Mono/Color Panel with Half Frame Buffer Disabled.⁵ 	800x600 ²	1/2/4/8	25	32	50	-
			16		56	48	-
		640x480	1/2/4/8		32	77	60
			16		56	75	60
		640x240	1/2/4/8		32	142	-
			16		56	136	-
		480x320	1/2/4/8		32	152	-
			16		56	145	-
		320x240	1/2/4/8		32	294	-
			16		56	280	-
	<ul style="list-style-type: none"> • Dual Mono with Half Frame Buffer Enabled. 	800x600 ²	1/2/4/8/16	12.5	32	50	-
			1/2/4/8/16	12.5	32	77	-
			1/2/4/8/16	12.5	32	92	-
		800x600 ^{2,3}	1/2/4/8	12.5	32	50	-
			16	8.33	32	33	-
			1/2/4/8	12.5	32	77	-
	<ul style="list-style-type: none"> • Dual Color with Half Frame Buffer Enabled. 	640x480	16	8.33	32	51	-

1. Must set NRC = 4MCLK. See REG[22h], Performance Enhancement Register.
2. 800x600 @ 16 bpp requires 2M bytes of display buffer for all display types.
3. 800x600 @ 8 bpp on a dual color panel requires 2M bytes of display buffer if the half frame buffer is enabled.

4. Optimum frame rates for panels range from 60Hz to 150Hz. If the maximum refresh rate is too high for a panel, MCLK should be reduced or PCLK should be divided down.
5. Half Frame Buffer disabled by REG[1Bh] bit 0.
6. When setting a horizontal resolution greater than 767 pixels, with a color depth of 15/16 bpp, the Memory Offset Registers (REG[16h], REG[17h]) must be set to a virtual horizontal pixel resolution of 1024.

14.3 Bandwidth Calculation

When calculating the average bandwidth, there are two periods that must be calculated separately.

The first period is the time when the CPU is in competition with the display refresh fetches. The CPU can only access the memory when the display refresh releases the memory controller. The CPU bandwidth during this period is called the “bandwidth during display period”.

The second period is the time when the CPU has full access to the memory, with no competition from the display refresh. The CPU bandwidth during this period is called the “bandwidth during non display period.”

To calculate the average bandwidth, calculate the percentage of time between display period and non display period. The percentage of display period is multiplied with the bandwidth during display period. The percentage of non display period is multiplied with the bandwidth during non display period. The two products are summed to provide the average bandwidth.

Bandwidth during non display period

Based on simulation, it requires a minimum of 12 MCLKs to service one, two byte, CPU access to memory. This includes all the internal handshaking and assumes that NRC is set to 4MCLKs and the wait state bits are set to 10b.

Bandwidth during non display period = $f(\text{MCLK}) / 6 \text{ Mb/s}$

Bandwidth during display period

The amount of time taken up by display refresh fetches is a function of the color depth, and the display type. Below is a table of the number of MCLKs required for various memory fetches to display 16 pixels. Assuming NRC = 4MCLKs.

Table 14-4 Number of MCLKs Required for Various Memory Access

Memory Access	Number of MCLKs
Half Frame Buffer, monochrome	7
Half Frame Buffer, color	11
Display @ 1 bpp	4
Display @ 2 bpp	5
Display @ 4 bpp	7
Display @ 8 bpp	11
Display @ 16 bpp	19
CPU	4

Table 14-5 Total # MCLKs Taken for Display Refresh

Display	MCLKs for Display Refresh				
	1 bpp	2 bpp	4 bpp	8 bpp	16 bpp
<ul style="list-style-type: none"> • Single Panel. • CRT. • Dual Monochrome/Color Panel with Half Frame Buffer Disabled. • Simultaneous CRT + Single Panel. • Simultaneous CRT + Dual Monochrome/Color Panel with Half Frame Buffer Disabled. 	4	5	7	11	19
<ul style="list-style-type: none"> • Dual Monochrome Panel with Half Frame Buffer Enabled. • Simultaneous CRT + Dual Monochrome Panel with Half Frame Buffer Enable. 	11	12	14	18	26
<ul style="list-style-type: none"> • Dual Color Panel with Half Frame Buffer Enabled. 	15	16	18	22	30

Bandwidth during display period = MIN (bandwidth during non display period, B/C/D)

where B = number of MCLKs left available for CPU access after every 16 pixels drawn

= $(f(\text{MCLK})/f(\text{PCLK}) * 16 - \text{Total MCLK for Display refresh})$, units in MCLKs 16 pixels

where C = number of MCLKs required to service 1 CPU access (2 bytes of data)

= 4, units in MCLKs/2 bytes

where D = time to draw 16 pixels

= $16 / f(\text{PCLK})$, units in 16 pixels

The minimum function limits the bandwidth to the bandwidth available during non display period should the display fetches constitute a small percentage of the overall memory activity.

For 16 bpp single panel/CRT/dual panel with half frame buffer disable, the number of MCLKs required to fetch 16 pixels when PCLK = MCLK exceeds 16. In this case, the display fetch does not allow any CPU access during the display period. CPU access can only be achieved during non display periods.

Average Bandwidth

All displays have a horizontal non display period, and a vertical non display period. The formula for calculating the percentage of non display period is as follows

Percentage of non display period = $(\text{HTOT} * \text{VTOT} - \text{WIDTH} * \text{HEIGHT}) / (\text{HTOT} * \text{VTOT})$

Percentage of non display period for CRT = $(800 * 525 - 640 * 480) / (800 * 525) = 26.6\%$

Percentage of non display period for single panel = $(680 * 482 - 640 * 480) / 680 * 482 = 6.2\%$

Percentage of non display period for dual panel = $(680 * 242 - 640 * 240) / 680 * 242 = 6.6\%$

Average Bandwidth =

Percentage of non display period * Bandwidth during non display period +

(1- Percentage of non display period) * Bandwidth during display period

Table 14-6 Theoretical Maximum Bandwidth M byte/sec, Cursor/Ink Disabled

DRAM Type ¹ (Speed Grade)	640x480 Display	Max. Pixel Clock (MHz)	Maximum Bandwidth (M byte/sec)				
			1 bpp	2 bpp	4 bpp	8 bpp	16 bpp
50ns EDO-DRAM MCLK = 40MHz	CRT. Simultaneous CRT + Single Panel. Simultaneous CRT + Dual Monochrome/Color Panel with Half Frame Buffer Disabled.	40	6.67	6.67	6.67	6.36	1.79
	Single Panel.	40	6.67	6.67	6.60	6.27	0.41
	Dual Monochrome/Color Panel with Half Frame Buffer Disabled.	20	6.67	6.67	6.67	6.67	6.67
	Dual Monochrome Panel with Half Frame Buffer Enabled.	40	6.27	5.11	-	-	-
		20	6.67	6.67	6.67	6.67	3.94
		13.3	6.67	6.67	6.67	6.67	6.67
	Simultaneous CRT + Dual Mono Panel with Half Frame Buffer Enable.	40	6.36	5.44	-	-	-
	Dual Color Panel with Half Frame Buffer Enabled.	20	6.67	6.67	6.27	6.27	-
		13.3	6.67	6.67	6.67	6.67	6.67
60ns EDO-DRAM MCLK = 33MHz	CRT. Simultaneous CRT + Single Panel. Simultaneous CRT + Dual Monochrome/Color Panel with Half Frame Buffer Disabled.	33	5.5	5.5	5.5	5.24	1.47
	Single Panel.	33	5.5	5.5	5.5	5.17	0.34
	Dual Monochrome/Color Panel with Half Frame Buffer Disabled.	16.5	5.5	5.5	5.5	5.5	5.5
	Dual Monochrome Panel with Half Frame Buffer Enabled.	33	5.17	4.21	-	-	-
		16.5	5.5	5.5	5.5	5.5	3.25
		11	5.5	5.5	5.5	5.5	5.5
	Simultaneous CRT + Dual Monochrome Panel with Half Frame Buffer Enable.	33	5.24	4.49	-	-	-
	Dual Color Panel with Half Frame Buffer Enabled.	16.5	5.5	5.5	5.5	5.17	-
		11	5.5	5.5	5.5	5.5	5.5
60ns FPM-DRAM MCLK = 25MHz	CRT. Simultaneous CRT + Single Panel. Simultaneous CRT + Dual Monochrome/Color Panel with Half Frame Buffer Disabled.	25	4.16	4.16	4.16	3.97	1.11
	Single Panel.	25	4.16	4.16	4.16	3.92	0.26
	Dual Monochrome/Color Panel with Half Frame Buffer Disabled.	12.5	4.16	4.16	4.16	4.16	4.16
	Dual Monochrome with Half Frame Buffer Enabled.	25	3.92	3.19	-	-	-
		12.5	4.16	4.16	4.16	4.16	2.46
		8.3	4.16	4.16	4.16	4.16	4.16
	Simultaneous CRT + Dual Monochrome Panel with Half Frame Buffer Enable.	25	3.97	3.40	-	-	-
	Dual Color Panel with Half Frame Buffer Enabled.	12.5	4.16	4.16	4.16	3.92	-
		8.33	4.16	4.16	4.16	4.16	4.16

15 POWER SAVE MODES

Three power save modes are incorporated into the S1D13505 to meet the important need for power reduction in the hand-held device market.

Table 15-1 Power Save Mode Function Summary

Function	Power Save Mode (PSM)			
	Normal (Active)	No Display LCD Enable = 0 CRT Enable = 0	Software Suspend	Hardware Suspend
Display Active?	Yes	No	No	No
Register Access Possible?	Yes	Yes	Yes	No
Memory Access Possible?	Yes	Yes	No	No
LUT Access Possible?	Yes	Yes	Yes	No

Table 15-2 Pin States in Power-Save Modes

Pins	Pin State			
	Normal (Active)	No Display LCD Enable = 0 CRT Enable = 0	Software Suspend	Hardware Suspend
LCD outputs	Active (LCD Enable = 1)	Forced Low ²	Forced Low ²	Forced Low ²
LCDPWR	On (LCD Enable = 1)	Off	Off	Off
DRAM outputs	Active	CBR Refresh only	Refresh only ¹	Refresh only ¹
CRT/DAC outputs	Active (CRT Enable = 1)	Disabled	Disabled	Disabled
Host Interface outputs	Active	Active	Active	Disabled

1. Refresh method is selectable by REG[1Ah]. Supported methods are CBR refresh, self-refresh or no refresh at all.
2. The FPF_{FRAME} and FPL_{LINE} signals are set to their inactive states during power-down. The inactive states are determined by REG[07h] bit 6 and REG[0Ch] bit 6. A problem may occur if the inactive state is high (typical TFT/D-TFD configuration) and power is removed from the LCD panel.

For software suspend the problem can be solved in the following manner. At power-down, first enable software suspend, then wait ~120 V_{NDP}, and lastly reverse the polarity bits. At power-up, first disable software suspend, then revert the polarity bits back to the configuration state.

For hardware suspend an external hardware solution would be to use an AND gate on the sync signal. One input of the AND gate is connected to a sync signal, the other input would be tied to the panel's logic power supply. When the panel's logic power supply is removed, the sync signal is forced low.

16 MECHANICAL DATA

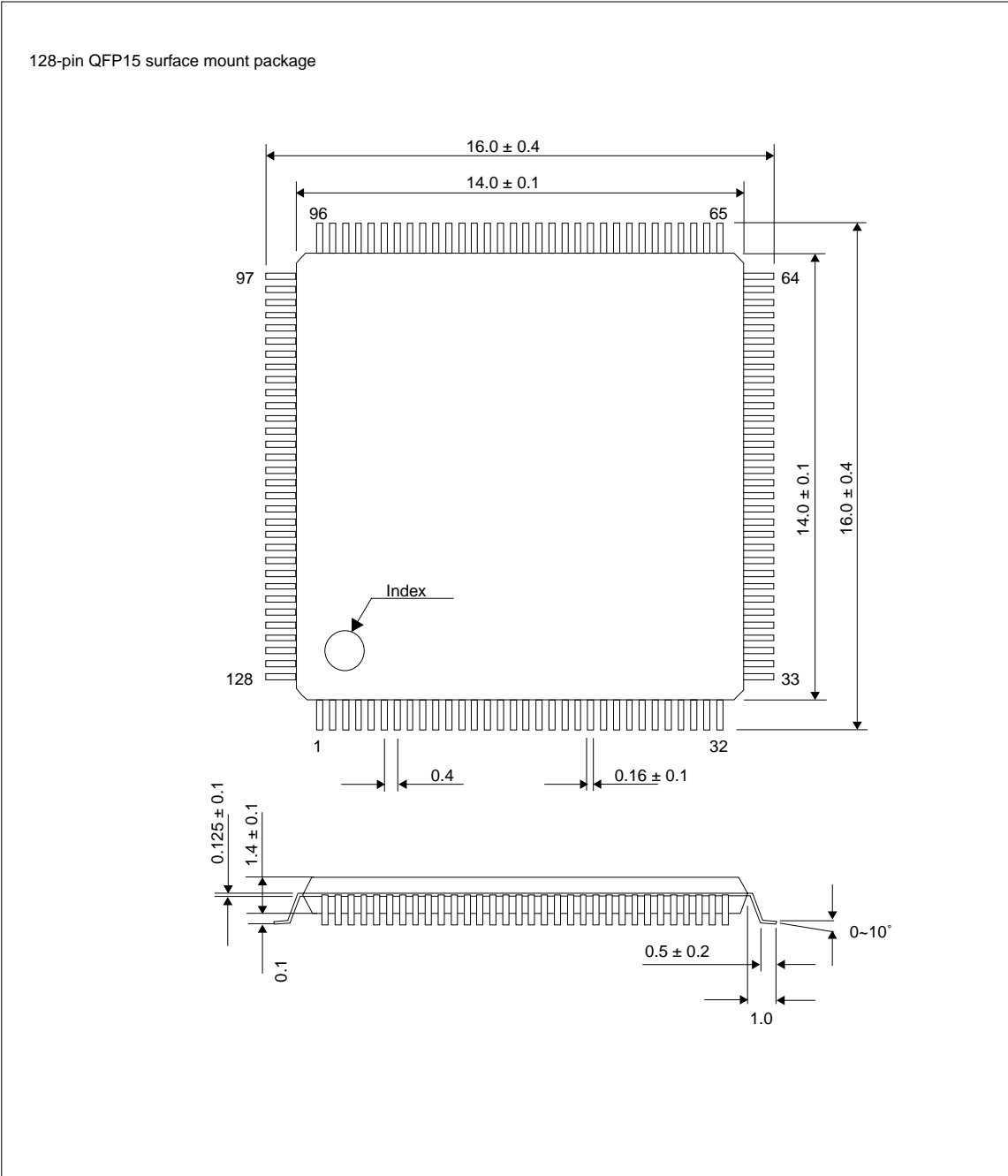


Figure 16-1 Mechanical Drawing QFP15

S1D13505F00A

Embedded RAMDAC LCD/CRT Controller

**Programming Notes
and Examples**

Table of Contents

1	INTRODUCTION	1
2	INITIALIZATION.....	2
2.1	Miscellaneous.....	4
3	MEMORY MODELS	5
3.1	Display Buffer Location	5
	Memory Organization for One Bit-Per-Pixel (2 Colors/Gray Shades)	5
	Memory Organization for Two Bit-Per-Pixel (4 Colors/Gray Shades)	5
	Memory Organization for Four Bit-Per-Pixel (16 Colors/Gray Shades)	6
	Memory Organization for Eight Bit-Per-Pixel (256 Colors/16 Gray Shades).....	6
	Memory Organization for Fifteen Bit-Per-Pixel (32768 Colors/16 Gray Shades).....	7
	Memory Organization for Sixteen Bit-Per-Pixel (65536 Colors/16 Gray Shades).....	7
4	LOOK-UP TABLE (LUT)	8
4.1	Look-Up Table Registers.....	8
4.2	Look-Up Table Organization	9
5	ADVANCED TECHNIQUES	15
5.1	Virtual Display	15
	Registers.....	16
	Examples	16
5.2	Panning and Scrolling	17
	Registers.....	18
	Examples	19
5.3	Split Screen	20
	Registers.....	20
	Examples	21
6	LCD POWER SEQUENCING AND POWER SAVE MODES	22
6.1	Introduction to LCD Power Sequencing	22
6.2	Registers	22
6.3	LCD Enable/Disable	23
7	HARDWARE CURSOR	24
7.1	Introduction.....	24
7.2	Registers	24
7.3	Limitations	26
	REG[29h] and REG[2Bh]	26
	REG[30h]	26
	No Top/Left Clipping on Hardware Cursor	26
7.4	Examples.....	26
8	HARDWARE ROTATION.....	27
8.1	Introduction to Hardware Rotation.....	27
8.2	S1D13505 Hardware Rotation	27
8.3	Registers	28
8.4	Limitations	29
8.5	Examples.....	29
9	CRT CONSIDERATIONS.....	31
9.1	Introduction.....	31
	CRT Only	31
	Simultaneous Display	31
10	IDENTIFYING THE S1D13505	32
11	HARDWARE ABSTRACTION LAYER (HAL).....	33
11.1	Introduction.....	33

11.2 API for 13505HAL	33
Initialization	33
General HAL Support	36
Advanced HAL Functions	39
Register / Memory Access	41
Color Manipulation	43
Drawing	45
Hardware Cursor	47
Ink Layer	50
Power Save	53
X-LIB Support	54
12 SAMPLE CODE	55
12.1 Introduction	55
Sample Code Using the 13505HAL API	55
Sample Code Without Using the 13505HAL API	57
Header Files	63
APPENDIX SUPPORTED PANEL VALUES	70

List of Figures

Figure 3-1	Pixel Storage for 1 Bpp (2 Colors/Gray Shades) in One Byte of Display Buffer	5
Figure 3-2	Pixel Storage for 2 Bpp (4 Colors/Gray Shades) in One Byte of Display Buffer	5
Figure 3-3	Pixel Storage for 4 Bpp (16 Colors/Gray Shades) in One Byte of Display Buffer	6
Figure 3-4	Pixel Storage for 8 Bpp (256 Colors/16 Gray Shades) in One Byte of Display Buffer	6
Figure 3-5	Pixel Storage for 15 Bpp (32768 Colors/16 Gray Shades) in Two Bytes of Display Buffer.....	7
Figure 3-6	Pixel Storage for 16 Bpp (65536 Colors/16 Gray Shades) in Two Bytes of Display Buffer.....	7
Figure 5-1	Viewport Inside a Virtual Display	15
Figure 5-2	Memory Address Offset Registers	16
Figure 5-3	Screen 1 Start Address Registers	18
Figure 5-4	Pixel Panning Register	18
Figure 5-5	320x240 Single Panel for Split Screen	20
Figure 5-6	Screen 1 Line Compare	20
Figure 5-7	Screen 2 Display Start Address	21

List of Tables

Table 2-1	S1D13505 Initialization Sequence	3
Table 4-1	Look-Up Table Configurations	9
Table 4-2	Recommended LUT Values for 1 Bpp Color Mode	9
Table 4-3	Example LUT Values for 2 Bpp Color Mode	10
Table 4-4	Suggested LUT Values to Simulate VGA Default 16 Color Palette.....	10
Table 4-5	Suggested LUT Values to Simulate VGA Default 256 Color Palette.....	11
Table 4-6	Recommended LUT Values for 1 Bpp Gray Shade	12
Table 4-7	Suggested Values for 2 Bpp Gray Shade	13
Table 4-8	Suggested LUT Values for 4 Bpp Gray Shade.....	13
Table 5-1	Number of Pixels Panned Using Start Address.....	18
Table 5-2	Active Pixel Pan Bits	18
Table 7-1	Ink/Cursor Mode	24
Table 7-2	Cursor/Ink Start Address Encoding	26
Table A-1	Passive Single Panel with 40MHz Pixel Clock	70
Table A-2	Passive Dual Panel with 40MHz Pixel Clock.....	71
Table A-3	TFT Single Panel with 25.175MHz Pixel Clock	71

1 INTRODUCTION

This guide demonstrates how to program the S1D13505 Embedded RAMDAC LCD/CDT Controller. The guide presents the basic concepts of the LCD/CRT controller and provides methods to directly program the registers. It explains some of the advanced techniques used and the special features of the S1D13505.

The guide also introduces the Hardware Abstraction Layer (HAL), which is designed to make programming the S1D13505 as easy as possible. Future S1D1350x products will support the HAL allowing OEMs the ability to upgrade to future chips with relative ease.

2 *INITIALIZATION*

S1D13505 initialization can be broken into three steps. First, enable the S1D13505 controller (if necessary identify the specific controller). Next, set all the registers to their initial values. Finally, program the Look-Up Table (LUT) with color values. This section does not deal with programming the LUT, see Section 4 of this manual for LUT programming details.

Note: When using an ISA evaluation board in a PC (i.e. S5U13505P00C), there are two additional steps that must be carried out before initialization. First, confirm that 16-bit mode is enabled by writing to address F80000h. Then, if hardware suspend is enabled, disable suspend mode by writing to F00000h. For further information on ISA evaluation boards refer to the *S5U13505P00C Rev. 1.0 ISA Bus Evaluation Board User Manual*.

The following table represents the sequence and values written to the S1D13505 registers to control a configuration with these specifications:

- 640x480 color dual passive format 1 LCD @ 75Hz.
- 8-bit data interface.
- 8 bit-per-pixel (bpp) - 256 colors.
- 31.5 MHz input clock.
- 50 ns EDO-DRAM, 2 CAS, 4 ms refresh, CAS before RAS.

Table 2-1 S1D13505 Initialization Sequence

Register	Value	Notes	See Also
[1B]	0000 0000	Enable the host interface	
[23]	1000 0000	Disable the FIFO	
[01]	0011 0000	Memory configuration - divide Clk1 by 512 to get 4 ms for 256 refresh cycles - this is 2-CAS# EDO memory	
[22]	0100 1000	Performance Enhancement 0 - refer to the hardware specification for a complete description of these bits	S1D13505 Hardware Functional Specification
[02]	0001 0110	Panel type - non-EL, 8-bit data, format 1, color, dual, passive	
[03]	0000 0000	Mod rate used by older monochrome panels - set to 0	
[04]	0100 1111	Horizontal display size = (REG[04]+1)*8 = (79+1)*8 = 640 pixels	see note for REG[16h] and REG[17h]
[05]	0000 0011	Horizontal non-display size = (REG[05]+1)*8 = (3+1)*8 = 32 pixels	
[06]	0000 0000	FPLINE start position - only required for CRT or TFT/D-TFD	
[07]	0000 0000	FPLINE polarity set to active high	
[08]	1110 1111	Vertical display size = REG[09][08] + 1	
[09]	0000 0000	= 0000 0000 1110 1111 + 1 = 239+1 = 240 lines (total height/2 for dual panels)	
[0A]	0011 1000	Vertical non-display size = REG[0A] + 1 = 57 + 1 = 58 lines	
[0B]	0000 0000	FPFRAME start position - only required for CRT or TFT/D-TFD	
[0C]	0000 0000	FPFRAME polarity set to active high	
[0D]	0000 1100	Display mode - SwivelView™ disabled, 8 bpp and LCD disabled, enable LCD in last step of this example.	
[0E]	1111 1111	Line compare (REG[0Eh] and REG[0Fh] set to maximum allowable value. We can change this later if we want a split screen.	
[0F]	0000 0011		
[10]	0000 0000	Screen 1 Start Address (REG[10h], REG[11h], and REG[12h]) set to 0.	
[11]	0000 0000	This will start the display in the first byte of the display buffer.	
[12]	0000 0000		
[13]	0000 0000	Screen 2 Start Address (REG[13h], REG[14h], and REG[15h]) to offset	
[14]	0000 0000	0. Screen 2 Start Address in not used at this time.	
[15]	0000 0000		
[16]	0100 0000	Memory Address Offset (REG[17h], REG[16h])	
[17]	0000 0001	- 640 pixels = 640 bytes = 320 words = 140h words Note: When setting a horizontal resolution greater than 767 pixels, with a color depth of 15/16 bpp, the Memory Offset Registers (REG[16h], REG[17h]) must be set to a virtual horizontal pixel resolution of 1024.	
[18]	0000 0000	Set pixel panning for both screens to 0	
[19]	0000 0001	Clock Configuration - set PClk to MClk/2 - the specification says that for a dual color panel the maximum PClk is MClk/2	
[1A]	0000 0000	Enable LCD Power	
[1C]	0000 0000	MD Configuration Readback - we write a 0 here to keep the register configuration logic simpler	
[1D]	0000 0000	General I/O Pins - set to zero.	
[1E]	0000 0000		
[1F]	0000 0000		
[20]	0000 0000	General I/O Pins Control - set to zero.	
[21]	0000 0000		
[24]	0000 0000	The remaining register control operation of the LUT and hardware cursor/ink layer. During the chip initialization none of these registers needs to be set. It is safe to write them to zero as this is the power-up value for the registers.	
[26]	0000 0000		
[27]	0000 0000		
[28]	0000 0000		
[29]	0000 0000		
[2A]	0000 0000		
[2B]	0000 0000		
[2C]	0000 0000		
[2D]	0000 0000		
[2E]	0000 0000		
[2F]	0000 0000		
[30]	0000 0000		
[31]	0000 0000		
[23]	0000 0000	Enable FIFO, mask in appropriate FIFO threshold bits	S1D13505 Hardware Functional Specification
[0D]	0000 1101	Display mode - SwivelView™ disabled, 8 bpp and LCD enabled	

2.1 *Miscellaneous*

This section of the notes contains recommendations which can be set at initialization time to improve display image quality.

At high color depths the display FIFO introduces two conditions which must be accounted for in software. Simultaneous display while using a dual passive panel introduces another possible register change.

Display FIFO Threshold

At 15/16 bit-per-pixel the display FIFO threshold (bits 0–4 of REG[23h]) must be programmed to a value other than '0'. Product testing has shown that at these color depths a better quality image results when the display FIFO threshold is set to a value of 1Bh.

Memory Address Offset

When an 800x600 display mode is selected at 15 or 16 bpp, memory page breaks can disrupt the display buffer fetches. This disruption produces a visible flicker on the display. To avoid this set the Memory Address Offset (REG[16h] and REG[17h]) to 200h. This sets a 1024 pixel line which aligns the memory page breaks and reduces any flicker.

Half Frame Buffer Disable

The half frame buffer is an S1D13505 mechanism which pre-digitizes display data for dual panel displays. However, for proper simultaneous display operation the half frame buffer (HFB) must be disabled. When running simultaneous display with a dual panel the pattern used by the Frame Rate Modulator may need to be adjusted. This can be accomplished using the Alternate FRM Register Reg[31h]. In this case, the recommended value for REG[31h] of FFh, may produce more visually appealing output. For further information on the half frame buffer and the Alternate FRM Register see the “*S1D13505 Hardware Functional Specification*”.

3 MEMORY MODELS

The S1D13505 is capable of several color depths. The memory model for each color depth is packed pixel. Packed pixel data changes with each color depth from one byte containing eight consecutive pixels up to two bytes being required for one pixel.

3.1 Display Buffer Location

The S1D13505 requires either a 512K byte or 2M byte block of memory to be decoded by the system. System logic will determine the location of this memory block. See Section 9 of the “*Hardware Functional Specification*” for details.

Memory Organization for One Bit-Per-Pixel (2 Colors/Gray Shades)

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Pixel 0 Bit 1	Pixel 1 Bit 0	Pixel 2 Bit 1	Pixel 3 Bit 0	Pixel 4 Bit 1	Pixel 5 Bit 0	Pixel 6 Bit 1	Pixel 7 Bit 0

Figure 3-1 Pixel Storage for 1 Bpp (2 Colors/Gray Shades) in One Byte of Display Buffer

In this memory format each byte of display buffer contains eight adjacent pixels. Setting or resetting any pixel will require reading the entire byte, masking out the appropriate bits and, if necessary, setting the bits to ‘1’.

One bit pixels provide two gray shade/color possibilities. For monochrome panels the two gray shades are generated by indexing into the first two elements of the green component of the Look-Up Table (LUT). For color panels the two colors are derived by indexing into positions 0 and 1 of the Look-Up Table.

Memory Organization for Two Bit-Per-Pixel (4 Colors/Gray Shades)

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Pixel 0 Bit 1	Pixel 0 Bit 0	Pixel 1 Bit 1	Pixel 1 Bit 0	Pixel 2 Bit 1	Pixel 2 Bit 0	Pixel 3 Bit 1	Pixel 3 Bit 0

Figure 3-2 Pixel Storage for 2 Bpp (4 Colors/Gray Shades) in One Byte of Display Buffer

In this memory format each byte of display buffer contains four adjacent pixels. Setting or resetting any pixel will require reading the entire byte, masking out the appropriate bits and, if necessary, setting the bits to ‘1’.

Two bit pixels are capable of displaying four gray shade/color combinations. For monochrome panels the four gray shades are generated by indexing into the first four elements of the green component of the Look-Up Table. For color panels the four colors are derived by indexing into positions 0 through 3 of the Look-Up Table.

Memory Organization for Four Bit-Per-Pixel (16 Colors/Gray Shades)

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Pixel 0 Bit 3	Pixel 0 Bit 2	Pixel 0 Bit 1	Pixel 0 Bit 0	Pixel 1 Bit 3	Pixel 1 Bit 2	Pixel 1 Bit 1	Pixel 1 Bit 0

Figure 3-3 Pixel Storage for 4 Bpp (16 Colors/Gray Shades) in One Byte of Display Buffer

In this memory format each byte of display buffer contains two adjacent pixels. Setting or resetting any pixel will require reading the entire byte, masking out the upper or lower nibble (4 bits) and setting the appropriate bits to '1'.

Four bit pixels provide 16 gray shade/color possibilities. For monochrome panels the gray shades are generated by indexing into the first 16 elements of the green component of the Look-Up Table. For color panels the 16 colors are derived by indexing into the first 16 positions of the Look-Up Table.

Memory Organization for Eight Bit-Per-Pixel (256 Colors/16 Gray Shades)

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
One Pixel							

Figure 3-4 Pixel Storage for 8 Bpp (256 Colors/16 Gray Shades) in One Byte of Display Buffer

In eight bit-per-pixel mode each byte of display buffer represents one pixel on the display. At this color depth the read-modify-write cycles of the lessor pixel depths are eliminated.

Each byte indexes into one of the 256 positions of the Look-Up Table. The S1D13505 LUT supports four bits per primary color, therefore this translates into 4096 possible colors when color mode is selected. To display the fullest dynamic range of colors will require careful selection of the colors in the LUT indices and in the image to be displayed.

When monochrome mode is selected, the green component of the LUT is used to determine the gray shade intensity. The green indices, with only four bits, can resolve 16 gray shades. In this situation one might as well use four bit-per-pixel mode and conserve display buffer.

Memory Organization for Fifteen Bit-Per-Pixel (32768 Colors/16 Gray Shades)

Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
Reserved	Red Bit 4	Red Bit 3	Red Bit 2	Red Bit 1	Red Bit 0	Green Bit 4	Green Bit 3
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Green Bit 2	Green Bit 1	Green Bit 0	Blue Bit 4	Blue Bit 3	Blue Bit 2	Blue Bit 1	Blue Bit 0

Figure 3-5 Pixel Storage for 15 Bpp (32768 Colors/16 Gray Shades) in Two Bytes of Display Buffer

In 15 bit-per-pixel mode the S1D13505 is capable of displaying 32768 colors. The 32768 color pixel is divided into four parts: one reserved bit, five bits for red, five bits for green, and five bits for blue. In this mode the Look-Up Table is bypassed and output goes directly into the Frame Rate Modulator.

The full color range is only available on TFT/D-TFD or CRT displays. Passive LCD displays are limited to using the four most significant bits from each of the red, green and blue portions of each color. The result is 4096 ($2^4 * 2^4 * 2^4$) possible colors.

Should monochrome mode be chosen at this color depth, the output reverts to sending the four most significant bits of the green LUT component to the modulator for a total of 16 possible gray shades. In this situation one might as well use four bit-per-pixel mode and conserve display buffer.

Memory Organization for Sixteen Bit-Per-Pixel (65536 Colors/16 Gray Shades)

Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
Red Bit 4	Red Bit 3	Red Bit 2	Red Bit 1	Red Bit 0	Green Bit 5	Green Bit 4	Green Bit 3
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Green Bit 2	Green Bit 1	Green Bit 0	Blue Bit 4	Blue Bit 3	Blue Bit 2	Blue Bit 1	Blue Bit 0

Figure 3-6 Pixel Storage for 16 Bpp (65536 Colors/16 Gray Shades) in Two Bytes of Display Buffer

In 16 bit-per-pixel mode the S1D13505 is capable of generating 65536 colors. The 65536 color pixel is divided into three parts: five bits for red, six bits for green, and five bits for blue. In this mode the Look-Up Table is bypassed and output goes directly into the Frame Rate Modulator.

The full color range is only available on TFT/D-TFD or CRT displays. Passive LCD displays are limited to using the four most significant bits from each of the red, green and blue portions of each color. The result is 4096 ($2^4 * 2^4 * 2^4$) possible colors.

When monochrome mode is selected, the green component of the LUT is used to determine the gray shade intensity. The green indices, with only four bits, can resolve 16 gray shades. In this situation one might as well use four bit-per-pixel mode and conserve display buffer.

4 LOOK-UP TABLE (LUT)

This section is supplemental to the description of the Look-Up Table architecture found in the “*S1D13505 Hardware Functional Specification*”. Covered here is a review of the LUT registers, recommendations for the color and gray shade LUT values, and additional programming considerations for the LUT. Refer to the “*S1D13505 Hardware Functional Specification*” for more detail.

The S1D13505 Look-Up Table is used for both the CRT and panel interface and consists of 256 indexed red/green/blue entries. Each entry is 4 bits wide. Two registers, at offsets 0x24 and 0x26, control access to the LUT. Color depth affects how many indices will be used for image display.

In color modes, pixel values are used as indices to an RGB value stored in the Look-Up Table. In monochrome modes only the green component of the LUT is used. The value in the display buffer indexes into the LUT and the amount of green at that index controls the intensity. Monochrome mode look-ups are done for the panel interface only. The CRT interface always receives the RGB values from the Look-Up Table.

4.1 Look-Up Table Registers

REG[24h] Look-Up Table Address Register							Read/Write
LUT Address Bit 7	LUT Address Bit 6	LUT Address Bit 5	LUT Address Bit 4	LUT Address Bit 3	LUT Address Bit 2	LUT Address Bit 1	LUT Address Bit 0

LUT Address

The LUT address register selects which of the 256 LUT entries will be accessed. Writing to this register will select the red bank. After three successive reads or writes to the data register this register will be incremented by one.

REG[24h] Look-Up Table Address Register							Read/Write
LUT Data Bit 3	LUT Data Bit 2	LUT Data Bit 1	LUT Data Bit 0	n/a	n/a	n/a	n/a

LUT Data

This register is where the 4-bit red/green/blue data value is written or read. With each successive read or write the internal bank select is incremented. Three reads from this register will result in reading the red, then the green, and finally the blue values associated with the index set in the LUT address register.

After the third read the LUT address register is incremented and the internal index points to the red bank again.


4.2 Look-Up Table Organization

- The Look-Up Table treats the value of a pixel as an index into an array of colors or gray shades. For example, a pixel value of zero would point to the first LUT entry; a pixel value of 7 would point to the eighth LUT entry.
- The value inside each LUT entry represents the intensity of the given color or gray shade. This intensity can range in value between 0 and 0Fh.
- The S1D13505 Look-Up Table is linear; increasing the LUT entry number results in a lighter color or gray shade. For example, a LUT entry of 0Fh into the red LUT entry will result in a bright red output while a LUT entry of 5 would result in a dull red.

Table 4-1 Look-Up Table Configurations

Display Mode	4-Bit Wide Look-Up Table			Effective Gray Shade/Colors on an Passive Panel
	RED	GREEN	BLUE	
1 bpp gray		2		2 gray shades
2 bpp gray		4		4 gray shades
4 bpp gray		16		16 gray shades
8 bpp gray		16		16 gray shades
15 bpp gray				16 gray shades
16 bpp gray				16 gray shades
1 bpp color	2	2	2	2 colors
2 bpp color	4	4	4	4 colors
4 bpp color	16	16	16	16 colors
8 bpp color	256	256	256	256 colors
15 bpp color				4096 colors*
16 bpp color				4096 colors*

* On an active matrix panel the effective colors are determined by the interface width. (i.e. 9-bit=512, 12-bit=4096, 18-bit=64K colors) Passive panels are limited to 12-bits through the Frame Rate Modulator.

 Indicates the Look-Up Table is not used for that display mode.

Color Modes

In color display modes, depending on the color depth, 2 through 256 index entries are used. The selection of which entries are used is automatic.

1 bpp Color


When the S1D13505 is configured for 1 bpp color mode, the LUT is limited to the first two entries. The two LUT entries can be any two RGB values but are typically set to black-and-white.

Each byte in the display buffer contains 8 bits, each pertaining to adjacent pixels. A bit value of '0' results in the LUT 0 index value being displayed. A bit value of '1' results in the LUT 1 index value being displayed.

The following table shows the recommended values for obtaining a black-and-white mode while in 1 bpp on a color panel.

Table 4-2 Recommended LUT Values for 1 Bpp Color Mode

Index	Red	Green	Blue
00	00	00	00
01	F0	F0	F0
02	00	00	00
...	00	00	00
FF	00	00	00

 Indicates unused entries in the LUT.

2 bpp Color

When the S1D13505 is configured for 2 bpp color mode only the first 4 entries of the LUT are used. These four entries can be set to any desired values.

Each byte in the display buffer contains 4 adjacent pixels. Each pair of bits in the byte are used as an index into the LUT. The following table shows example values for 2 bpp color mode.

Table 4-3 Example LUT Values for 2 Bpp Color Mode

Index	Red	Green	Blue
00	00	00	00
01	70	70	70
02	A0	A0	A0
03	F0	F0	F0
04	00	00	00
...	00	00	00
FF	00	00	00



Indicates unused entries in the LUT.

4 bpp Color

When the S1D13505 is configured for 4 bpp color mode the first 16 entries in the LUT are used.

Each byte in the display buffer contains two adjacent pixels. The upper and lower nibbles of the byte are used as indices into the LUT.

The following table shows LUT values that will simulate those of a VGA operating in 16 color mode.

Table 4-4 Suggested LUT Values to Simulate VGA Default 16 Color Palette

Index	Red	Green	Blue
00	00	00	00
01	00	00	0A
02	00	0A	00
03	00	0A	0A
04	0A	00	00
05	0A	00	0A
06	0A	0A	00
07	0A	0A	0A
08	00	00	00
09	00	00	0F
0A	00	0F	00
0B	00	0F	0F
0C	0F	00	00
0D	0F	00	0F
0E	0F	0F	00
0F	0F	0F	0F
10	00	00	00
...	00	00	00
FF	00	00	00



Indicates unused entries in the LUT.

8 bpp Color

When the S1D13505 is configured for 8 bpp color mode all 256 entries in the LUT are used. Each byte in display buffer corresponds to one pixel and is used as an index value into the LUT.

The S1D13505 LUT has four bits (16 intensities) of intensity control per primary color while a standard VGA RAMDAC has six bits (64 intensities). This four to one difference has to be considered when attempting to match colors between a VGA RAMDAC and the S1D13505 LUT. (i.e. VGA levels 0 - 3 map to LUT level 0, VGA levels 4 - 7 map to LUT level 1...). Additionally, the significant bits of the color tables are located at different offsets within their respective bytes. After calculating the equivalent intensity value the result must be shifted into the correct bit positions.

The following table shows LUT values that will approximate the VGA default color palette.

Table 4-5 Suggested LUT Values to Simulate VGA Default 256 Color Palette

Index	R	G	B	Index	R	G	B	Index	R	G	B	Index	R	G	B
00	00	00	00	40	F0	70	70	80	30	30	70	C0	00	40	00
01	00	00	A0	41	F0	90	70	81	40	30	70	C1	00	40	10
02	00	A0	00	42	F0	B0	70	82	50	30	70	C2	00	40	20
03	00	A0	A0	43	F0	D0	70	83	60	30	70	C3	00	40	30
04	A0	00	00	44	F0	F0	70	84	70	30	70	C4	00	40	40
05	A0	00	A0	45	D0	F0	70	85	70	30	60	C5	00	30	40
06	A0	50	00	46	B0	F0	70	86	70	30	50	C6	00	20	40
07	A0	A0	A0	47	90	F0	70	87	70	30	40	C7	00	10	40
08	50	50	50	48	70	F0	70	88	70	30	30	C8	20	20	40
09	50	50	F0	49	70	F0	90	89	70	40	30	C9	20	20	40
0A	50	F0	50	4A	70	F0	B0	8A	70	50	30	CA	30	20	40
0B	50	F0	F0	4B	70	F0	D0	8B	70	60	30	CB	30	20	40
0C	F0	50	50	4C	70	F0	F0	8C	70	70	30	CC	40	20	40
0D	F0	50	F0	4D	70	D0	F0	8D	60	70	30	CD	40	20	30
0E	F0	F0	50	4E	70	B0	F0	8E	50	70	30	CE	40	20	30
0F	F0	F0	F0	4F	70	90	F0	8F	40	70	30	CF	40	20	20
10	00	00	00	50	B0	B0	F0	90	30	70	30	D0	40	20	20
11	10	10	10	51	C0	B0	F0	91	30	70	40	D1	40	20	20
12	20	20	20	52	D0	B0	F0	92	30	70	50	D2	40	30	20
13	20	20	20	53	E0	B0	F0	93	30	70	60	D3	40	30	20
14	30	30	30	54	F0	B0	F0	94	30	70	70	D4	40	40	20
15	40	40	40	55	F0	B0	E0	95	30	60	70	D5	30	40	20
16	50	50	50	56	F0	B0	D0	96	30	50	70	D6	30	40	20
17	60	60	60	57	F0	B0	C0	97	30	40	70	D7	20	40	20
18	70	70	70	58	F0	B0	B0	98	50	50	70	D8	20	40	20
19	80	80	80	59	F0	C0	B0	99	50	50	70	D9	20	40	20
1A	90	90	90	5A	F0	D0	B0	9A	60	50	70	DA	20	40	30
1B	A0	A0	A0	5B	F0	E0	B0	9B	60	50	70	DB	20	40	30
1C	B0	B0	B0	5C	F0	F0	B0	9C	70	50	70	DC	20	40	40
1D	C0	C0	C0	5D	E0	F0	B0	9D	70	50	60	DD	20	30	40
1E	E0	E0	E0	5E	D0	F0	B0	9E	70	50	60	DE	20	30	40
1F	F0	F0	F0	5F	C0	F0	B0	9F	70	50	50	DF	20	20	40
20	00	00	F0	60	B0	F0	B0	A0	70	50	50	E0	20	20	40
21	40	00	F0	61	B0	F0	C0	A1	70	50	50	E1	30	20	40
22	70	00	F0	62	B0	F0	D0	A2	70	60	50	E2	30	20	40
23	B0	00	F0	63	B0	F0	E0	A3	70	60	50	E3	30	20	40
24	F0	00	F0	64	B0	F0	F0	A4	70	70	50	E4	40	20	40
25	F0	00	B0	65	B0	E0	F0	A5	60	70	50	E5	40	20	30
26	F0	00	70	66	B0	D0	F0	A6	60	70	50	E6	40	20	30
27	F0	00	40	67	B0	C0	F0	A7	50	70	50	E7	40	20	30
28	F0	00	00	68	00	00	70	A8	50	70	50	E8	40	20	20
29	F0	40	00	69	10	00	70	A9	50	70	50	E9	40	30	20
2A	F0	70	00	6A	30	00	70	AA	50	70	60	EA	40	30	20
2B	F0	B0	00	6B	50	00	70	AB	50	70	60	EB	40	30	20
2C	F0	F0	00	6C	70	00	70	AC	50	70	70	EC	40	40	20
2D	B0	F0	00	6D	70	00	50	AD	50	60	70	ED	30	40	20
2E	70	F0	00	6E	70	00	30	AE	50	60	70	EE	30	40	20
2F	40	F0	00	6F	70	00	10	AF	50	50	70	EF	30	40	20

Table 4-5 Suggested LUT Values to Simulate VGA Default 256 Color Palette (Continued)

Index	R	G	B	Index	R	G	B	Index	R	G	B	Index	R	G	B
30	00	F0	00	70	70	00	00	B0	00	00	40	F0	20	40	20
31	00	F0	40	71	70	10	00	B1	10	00	40	F1	20	40	30
32	00	F0	70	72	70	30	00	B2	20	00	40	F2	20	40	30
33	00	F0	B0	73	70	50	00	B3	30	00	40	F3	20	40	30
34	00	F0	F0	74	70	70	00	B4	40	00	40	F4	20	40	40
35	00	B0	F0	75	50	70	00	B5	40	00	30	F5	20	30	40
36	00	70	F0	76	30	70	00	B6	40	00	20	F6	20	30	40
37	00	40	F0	77	10	70	00	B7	40	00	10	F7	20	30	40
38	70	70	F0	78	00	70	00	B8	40	00	00	F8	00	00	00
39	90	70	F0	79	00	70	10	B9	40	10	00	F9	00	00	00
3A	B0	70	F0	7A	00	70	30	BA	40	20	00	FA	00	00	00
3B	D0	70	F0	7B	00	70	50	BB	40	30	00	FB	00	00	00
3C	F0	70	F0	7C	00	70	70	BC	40	40	00	FC	00	00	00
3D	F0	70	D0	7D	00	50	70	BD	30	40	00	FD	00	00	00
3E	F0	70	B0	7E	00	30	70	BE	20	40	00	FE	00	00	00
3F	F0	70	90	7F	00	10	70	BF	10	40	00	FF	00	00	00

15 bpp color

The Look-Up Table is bypassed at this color depth, hence programming the LUT is not necessary.

16 bpp color

The Look-Up Table is bypassed at this color depth, hence programming the LUT is not necessary.

Gray Shade Modes

This discussion of gray shade/monochrome modes only applies to the panel interface. Monochrome mode is selected when register [01] bit 2 = 0. In this mode the output value to the panel is derived solely from the green component of the LUT. The CRT image will continue to be formed from all three (RGB) Look-Up Table components.

Note: In order to match the colors on a CRT with the colors on a monochrome panel it is important to ensure that the red and blue components of the Look-Up Table be set to the same intensity as the green component.

1 bpp gray shade

In 1 bpp gray shade mode only the first two entries of the green LUT are used. All other LUT entries are unused.

Table 4-6 Recommended LUT Values for 1 Bpp Gray Shade

Address	Red	Green	Blue
00	00	00	00
01	F0	F0	F0
02	00	00	00
...	00	00	00
FF	00	00	00



Required to match CRT to panel



Unused entries

2 bpp gray shade

In 2 bpp gray shade mode the first four green elements are used to provide values to the panel. The remaining indices are unused.

Table 4-7 Suggested Values for 2 Bpp Gray Shade

Index	Red	Green	Blue
0	00	00	00
1	50	50	50
2	A0	A0	A0
3	F0	F0	F0
4	00	00	00
...	00	00	00
FF	00	00	00



Required to match CRT to panel



Unused entries

4 bpp Gray Shade

The 4 bpp gray shade mode uses the first 16 LUT elements. The remaining indices of the LUT are unused.

Table 4-8 Suggested LUT Values for 4 Bpp Gray Shade

Index	Red	Green	Blue
00	00	00	00
01	10	10	10
02	20	20	20
03	30	30	30
04	40	40	40
05	50	50	50
06	60	60	60
07	70	70	70
08	80	80	80
09	90	90	90
0A	A0	A0	A0
0B	B0	B0	B0
0C	C0	C0	C0
0D	D0	D0	D0
0E	E0	E0	E0
0F	F0	F0	F0
10	00	00	00
...	00	00	00
FF	00	00	00



Required to match CRT to panel



Unused entries

8 bpp gray shade

When 8 bpp gray shade mode is selected the gray shade intensity is determined by the green LUT value. The green portion of the LUT has 16 possible intensities. There is no color advantage to selecting 8 bpp mode over 4 bpp mode; however, hardware rotate can be only used in 8 and 16 bpp modes.

15 bpp gray shade

The Look-Up Table is bypassed at this color depth, hence programming the LUT is not necessary.

As with 8 bpp there are limitations to the colors which can be displayed. In this mode the four most significant bits of green are used to set the absolute intensity of the image. Four bits of green resolves to 16 colors. Now however, each pixel requires two bytes.

16 bpp gray

The Look-Up Table is bypassed at this color depth, hence programming the LUT is not necessary.

As with 8 bpp there are limitations to the colors which can be displayed. In this mode the four most significant bits of green are used to set the absolute intensity of the image. Four bits of green resolves to 16 colors. Now however, each pixel requires two bytes.

5 ADVANCED TECHNIQUES

This section presents information on the following:

- virtual display
- panning and scrolling
- split screen display

5.1 Virtual Display

Virtual display refers to the situation where the image to be viewed is larger than the physical display. This can be in the horizontal, the vertical or both dimensions. To view the image, the display is used as a window (or viewport) into the display buffer. At any given time only a portion of the image is visible. Panning and scrolling are used to view the full image.

The Memory Address Offset registers are used to determine the number of horizontal pixels in the virtual image. The offset registers can be set for a maximum of 2^{11} or 2048 words. In 1 bpp display modes these 2048 words cover 16,384 pixels. At 16 bpp 2048 words cover 1024 pixels.

The maximum vertical size of the virtual image is the result of a number of variables. In its simplest, the number of lines is the total display buffer divided by the number of bytes per horizontal line. The number of bytes per line is the number of words in the offset register multiplied by two. At maximum horizontal size, the greatest number of lines that can be displayed is 1024. Reducing the horizontal size makes memory available to increase the virtual vertical size.

In addition to the calculated limit the virtual vertical size is limited by the size and location of the half frame buffer and the ink/cursor if present.

Seldom are the maximum sizes used. Figure 5-1 “Viewport Inside a Virtual Display,” depicts a more typical use of a virtual display. The display panel is 320x240 pixels, an image of 640x480 pixels can be viewed by navigating a 320x240 pixel viewport around the image using panning and scrolling.

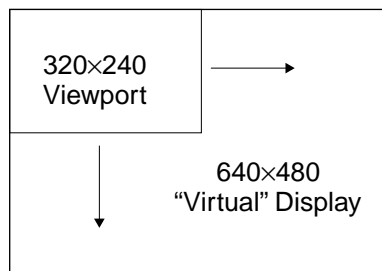


Figure 5-1 Viewport Inside a Virtual Display

Registers

REG[16h] Memory Address Offset Register 0							
Memory Address Offset Bit 7	Memory Address Offset Bit 6	Memory Address Offset Bit 5	Memory Address Offset Bit 4	Memory Address Offset Bit 3	Memory Address Offset Bit 2	Memory Address Offset Bit 1	Memory Address Offset Bit 0

REG[17h] Memory Address Offset Register 1							
n/a	n/a	n/a	n/a	n/a	Memory Address Offset Bit 10	Memory Address Offset Bit 9	Memory Address Offset Bit 8

Figure 5-2 Memory Address Offset Registers

Registers [16h] and [17h] form an 11-bit value called the memory address offset. This offset is the number of words from the beginning of one line of the display to the beginning of the next line of the display.

Note that this value does not necessarily represent the number of words to be shown on the display. The display width is set in the Horizontal Display Width register. If the offset is set to the same as the display width then there is no virtual width.

To maintain a constant virtual width as color depth changes, the memory address offset must also change. At 1 bpp each word contains 16 pixels, at 16 bpp each word contains one pixel. The formula to determine the value for these registers is:

$$\text{offset} = \text{pixels_per_line} / \text{pixels_per_word}$$

Examples

Example 1

Determine the offset value required for 800 pixels at a color depth of 8 bpp.

At 8 bpp each byte contains one pixel, therefore each word contains two pixels.

$$\text{pixels_per_word} = 16 / \text{bpp} = 16 / 8 = 2$$

Using the above formula.

$$\text{offset} = \text{pixels_per_line} / \text{pixels_per_word} = 800 / 2 = 400 = 0x190 \text{ words}$$

Register [17h] would be set to 0x01 and register [16h] would be set to 0x90.

Example 2

Program the Memory Address Offset Registers to support a 16 color (4 bpp) 640x480 virtual display on a 320x240 LCD panel.

To create a virtual display the offset registers must be programmed to the horizontal size of the larger “virtual” image. After determining the amount of memory used by each line, do a calculation to see if there is enough memory to support the desired number of lines.

1. Initialize the S1D13505 registers for a 320x240 panel. (See “Introduction” on page 1).
2. Determine the offset register value.

$$\text{pixels_per_word} = 16 / \text{bpp} = 16 / 4 = 4$$

$$\text{offset} = \text{pixels_per_line} / \text{pixels_per_word} = 640 / 4 = 160 \text{ words} = 0x0A0 \text{ words}$$

Register [17h] will be written with 0x00 and register [16h] will be written with 0xA0.

3. Check that we have enough memory for the required virtual height.
4. Each line uses 160 words and we need 480 lines for a total of (160×480) 76,800 words. This display could be done on a system with the minimum supported memory size of 512K bytes. It is safe to continue with these values.

5.2 *Panning and Scrolling*

The terms panning and scrolling refer to the actions used to move the viewport about a virtual display. Although the image is stored entirely in the display buffer, only a portion is actually visible at any given time.

Panning describes the horizontal (side to side) motion of the viewport. When panning to the right the image in the viewport appears to slide to the left. When panning to the left the image appears to slide to the right. Scrolling describes the vertical (up and down) motion of the viewport. Scrolling down causes the image to appear to slide up and scrolling up causes the image to appear to slide down.

Both panning and scrolling are performed by modifying the start address register. The start address refers to the word offset in the display buffer where the image will start being displayed from. At color depths less than 15 bpp a second register, the pixel pan register, is required for smooth pixel level panning.

Internally, the S1D13505 latches different signals at different times. Due to this internal sequence, there is an order in which the start address and pixel pan registers should be accessed during scrolling operations to provide the smoothest scrolling. Setting the registers in the wrong sequence or at the wrong time will result in a “tearing” or jitter effect on the display.

The start address is latched at the beginning of each frame, therefore the start address can be set any time during the display period. The pixel pan register values are latched at the beginning of each display line and must be set during the vertical non-display period. The correct sequence for programming these registers is:

1. Wait until just after a vertical non-display period (read register [0Ah] and watch bit 7 for the non-display status).
2. Update the start address registers.
3. Wait until the next vertical non-display period.
4. Update the pixel panning register.

Registers

REG[10h] Screen 1 Display Start Address 0							
Start Address Bit 7	Start Address Bit 6	Start Address Bit 5	Start Address Bit 4	Start Address Bit 3	Start Address Bit 2	Start Address Bit 1	Start Address Bit 0
REG[11h] Screen 1 Display Start Address 1							
Start Address Bit 15	Start Address Bit 14	Start Address Bit 13	Start Address Bit 12	Start Address Bit 11	Start Address Bit 10	Start Address Bit 9	Start Address Bit 8
REG[12h] Screen 1 Display Start Address 2							
n/a	n/a	n/a	n/a	Start Address Bit 19	Start Address Bit 18	Start Address Bit 17	Start Address Bit 16

Figure 5-3 Screen 1 Start Address Registers

These three registers form the address of the word in the display buffer where screen 1 will start displaying from. Changing these registers by one will cause a change of 0 to 16 pixels depending on the current color depth. Refer to the following table to see the minimum number of pixels affected by a change of one to these registers.

Table 5-1 Number of Pixels Panned Using Start Address

Color Depth (bpp)	Pixels Per Word	Number of Pixels Panned
1	16	16
2	8	8
4	4	4
8	2	2
15	1	1
16	1	1

REG[18h] Pixel Panning Register							
Screen 2 Pixel Pan Bit 3	Screen 2 Pixel Pan Bit 2	Screen 2 Pixel Pan Bit 1	Screen 2 Pixel Pan Bit 0	Screen 1 Pixel Pan Bit 3	Screen 1 Pixel Pan Bit 2	Screen 1 Pixel Pan Bit 1	Screen 1 Pixel Pan Bit 0

Figure 5-4 Pixel Panning Register

The pixel panning register offers finer control over pixel pans than is available with the Start Address Registers. Using this register it is possible to pan the displayed image one pixel at a time. Depending on the current color depth certain bits of the pixel pan register are not used. The following table shows this.

Table 5-2 Active Pixel Pan Bits

Color Depth (bpp)	Pixel Pan Bits Used
1	bits [3:0]
2	bits [2:0]
4	bits [1:0]
8	bit 0
15/16	---

Examples

For the examples in this section assume that the display system has been set up to view a 640x480 pixel image in a 320x240 viewport. Refer to Section 2, “*Initialization*” on page 2 and Section 5.1, “*Virtual Display*” on page 15 for assistance with these settings.

Example 3

Panning - Right and Left

To pan to the right, increment the pixel pan value. If the pixel pan value is equal to the current color depth then set the pixel pan value to zero and increment the start address value. To pan to the left decrement the pixel pan value. If the pixel pan value is less than zero set it to the color depth (bpp) less one and decrement the start address.

Note: Scrolling operations are easier to follow if a value, call it `pan_value`, is used to track both the pixel pan and start address. The least significant bits of `pan_value` will represent the pixel pan value and the more significant bits are the start address value.

The following pans to the right by one pixel in 4 bpp display mode.

1. This is a pan to the right. Increment `pan_value`.

```
pan_value = pan_value + 1
```

2. Mask off the values from `pan_value` for the pixel panning and start address register portions. In this case, 4 bpp, the lower two bits are the pixel panning value and the upper bits are the start address.

```
pixel_pan = pan_value AND 3
```

```
start_address = pan_value SHR 3 (the first two bits of the shift account for the pixel_pan
the last bit of the shift converts
the start_address value
from bytes to words)
```

3. Write the pixel panning and start address values to their respective registers using the procedure outlined in the registers section.

Example 4

Scrolling - Up and Down

To scroll down, increase the value in the Screen 1 Display Start Address Register by the number of words in one *virtual* scan line. To scroll up, decrease the value in the Screen 1 Display Start Address Register by the number of words in one *virtual* scan line.

Example 5

Scroll down one line for a 16 color 640x480 virtual image using a 320x240 single panel LCD.

1. To scroll down we need to know how many words each line takes up. At 16 colors (4 bpp) each byte contains two pixels so each word contains 4 pixels.

```
offset_words = pixels_per_line / pixels_per_word = 640 / 4 = 160 = 0xA0
```

We now know how much to add to the start address to scroll down one line.

2. Increment the start address by the number of words per virtual line.

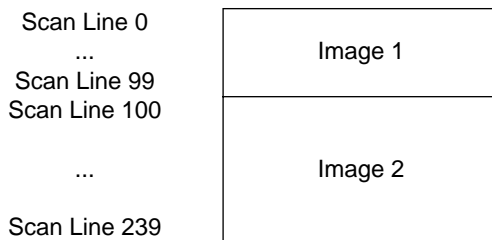
```
start_address = start_address + words
```

3. Separate the start address value into three bytes. Write the LSB to register [10h] and the MSB to register [12h].

5.3 Split Screen

Occasionally the need arises to display two distinct images on the display. For example, we may write a game where the main play area will rapidly update and we want a status display at the bottom of the screen.

The Split Screen feature of the S1D13505 allows a programmer to setup a display for such an application. The figure below illustrates setting a 320x240 panel to have Image 1 displaying from scan line 0 to scan line 99 and image 2 displaying from scan line 100 to scan line 239. Although this example picks specific values, image 1 and image 2 can be shown as varying portions of the screen.



Screen 1 Display Line Count Register = 99 lines

Figure 5-5 320x240 Single Panel for Split Screen

Registers

The other registers required for split screen operations, [10h] through [12h] (Screen 1 Display Start Address) and [18h] (Pixel Panning Register), are described in Section 5.2 on page 2-17.

REG[0E] Screen 1 Line Compare Register 0							
Line Compare Bit 7	Line Compare Bit 6	Line Compare Bit 5	Line Compare Bit 4	Line Compare Bit 3	Line Compare Bit 2	Line Compare Bit 1	Line Compare Bit 0

REG[0F] Screen 1 Line Compare Register 1							
n/a	n/a	n/a	n/a	n/a	n/a	Line Compare Bit 9	Line Compare Bit 8

Figure 5-6 Screen 1 Line Compare

These two registers form a value known as the line compare. When the line compare value is equal to or greater than the physical number of lines being displayed there is no visible effect on the display. When the line compare value is less than the number of physically displayed lines, display operation works like this:

1. From the end of vertical non-display to the number of lines indicated by line compare the display data will be from the memory pointed to by the Screen 1 Display Start Address.
2. After *line compare* lines have been displayed the display will begin showing data from Screen 2 Display Start Address memory.

REG[13h] Screen 2 Display Start Address Register 0							
Start Address Bit 7	Start Address Bit 6	Start Address Bit 5	Start Address Bit 4	Start Address Bit 3	Start Address Bit 2	Start Address Bit 1	Start Address Bit 0
REG[14h] Screen 2 Display Start Address Register 1							
Start Address Bit 15	Start Address Bit 14	Start Address Bit 13	Start Address Bit 12	Start Address Bit 11	Start Address Bit 10	Start Address Bit 9	Start Address Bit 8
REG[15h] Screen 2 Display Start Address Register 2							
n/a	n/a	n/a	n/a	Start Address Bit 19	Start Address Bit 18	Start Address Bit 17	Start Address Bit 16

Figure 5-7 Screen 2 Display Start Address

These three registers form the twenty bit offset to the first word in display buffer that will be shown in the screen 2 portion of the display.

Screen 1 memory is **always** the first memory displayed at the top of the screen followed by screen 2 memory. However, the start address for the screen 2 image may in fact be lower in memory than that of screen 1 (i.e. screen 2 could be coming from offset 0 in the display buffer while screen 1 was coming from an offset located several thousand bytes into display buffer). While not particularly useful, it is possible to set screen 1 and screen 2 to the same address.

Examples

Example 6

Display 380 scanlines of image 1 and 100 scanlines of image 2. Image 2 is located immediately after image 1 in the display buffer. Assume a 640x480 display and a color depth of 1 bpp.

1. The value for the line compare is not dependent on any other setting so we can set it immediately (380 = 0x17C).

Write the line compare registers [0Fh] with 0x01 and register [0Eh] with 0x7C.

2. Screen 1 is coming from offset 0 in the display buffer. Although not necessary, ensure that the screen 1 start address is set to zero.

Write 0x00 to registers [10h], [11h] and [12h].

3. Calculate the size of the screen 1 image (so we know where the screen 2 image is located). This calculation must be performed on the virtual size (offset register). Since a virtual size was not specified assume the virtual size to be the same as the physical size.

$\text{offset} = \text{pixels_per_line} / \text{pixels_per_word} = 640 / 16 = 40 \text{ words per line}$

$\text{screen1_size} = \text{offset} * \text{lines} = 40 * 480 = 19,200 \text{ words} = 0x4B00 \text{ words}$

4. Set the screen 2 start address to the value we just calculated.

Write the screen 2 start address registers [15h], [14h] and [13h] with the values 0x00, 0x4B and 0x00 respectively.

6 LCD POWER SEQUENCING AND POWER SAVE MODES

6.1 Introduction to LCD Power Sequencing

LCD Power Sequencing allows the LCD power supply to discharge prior to shutting down the LCD signals. Power sequencing is required to prevent long term damage to the panel and to avoid unsightly “lines” on power-down and power-up.

The S1D13505 performs automatic power sequencing when the LCD is enabled or disabled through the LCD Enable bit in register [0Dh]. For most applications the internal power sequencing is the appropriate choice.

There may be situations where the internal time delay is insufficient to discharge the LCD power supply before the LCD signals are shut down. This section details the sequences to manually power-up and power-down the LCD interface.

Proper LCD power sequencing dictates that there must be a time delay between the time the LCD power is disabled and the time the LCD signals are shut down. During power up the LCD signals must be active prior to applying power to the LCD. This time interval varies depending on the power supply design. The power supply on the S5U13505P00C Evaluation board requires 0.5 seconds to fully discharge. Your power supply design may vary.

6.2 Registers

REG[0D] Display Mode Register

SwivelView™ Enable	Simultaneous Display Option Select Bit 1	Simultaneous Display Option Select Bit 0	Bit-Per-Pixel Select Bit 2	Bit-Per-Pixel Select Bit 1	Bit-Per-Pixel Select Bit 0	CRT Enable	LCD Enable
-----------------------	--	--	-------------------------------	-------------------------------	-------------------------------	------------	------------

LCD Enable normally performs all the required power sequencing. Upon setting LCD Enable to '0' the system will begin a series of events which include turning off the LCD power supply, waiting for the power supply to discharge and finally turning off the LCD signals.

REG[1A] Power Save Configuration Register

Power Save Status	n/a	n/a	n/a	LCD Power Disable	Suspend Refresh Select Bit 1	Suspend Refresh Select Bit 0	Software Sus- pend Mode Enable
----------------------	-----	-----	-----	----------------------	------------------------------------	------------------------------------	--------------------------------------

LCD Power Disable would be used to manually sequence the events leading to an LCD power-down. First the program would set LCD Power Disable to '1' to begin discharging the LCD power supply. After waiting a pre-determined amount of time the software would Disable the LCD signals using the LCD Enable bit in register [0Dh].

6.3 *LCD Enable/Disable*

Power On/Enable Sequence

The following is the recommended sequence for manually powering-up an LCD panel. These steps would be used if power supply timing requirements were larger than the timings built into the S1D13505 power enable sequence.

1. Set REG[1Ah] bit 3 to 1. Ensure that LCD power is disabled.
2. Set REG[0Dh] bit 0 to 1. Turn on the LCD outputs.
3. Count 'x' Vertical Non-Display Periods.
'x' corresponds the power supply discharge time converted to the equivalent vertical non-display periods.
4. Set REG[1Ah] bit 3 to 0. This enables LCD Power.

Power On Sequence

The following is the recommended sequence for manually powering-down an LCD panel. These steps would be used if power supply timing requirements were larger than the timings built into the S1D13505 power disable sequence.

1. Set REG[1Ah] bit 3 to 1 - disable LCD Power.
2. Count 'x' Vertical Non-Display Periods.
'x' corresponds to the power supply discharge time converted to the equivalent vertical non-display periods.
3. Set REG[0Dh] bit 0 to 0 - turn off the LCD outputs.

7 *HARDWARE CURSOR*

7.1 *Introduction*

The S1D13505 provides hardware support for a cursor or an ink layer. These features are mutually exclusive and therefore only one or the other may be active at any given time.

A hardware cursor improves video throughput in graphical operating systems by off-loading much of the work typically assigned to software. Take the actions which must be performed when the user moves the mouse. On a system without hardware support, the operating system must restore the area under the current cursor position then save the area under the new location and finally draw the cursor shape. Contrast that with the hardware assisted system where the operating system must simply update the cursor X and cursor Y position registers.

An ink layer is used to support stylus or pen input. Without an ink layer the operating system would have to save an area (possibly all) of the display buffer where pen input was to occur. After the system recognized the user entered characters, the display would have to be restored and the characters redrawn in a system font. With an ink layer the stylus path is drawn in the ink layer, where it overlays the displayed image. After character recognition takes place the display is updated with the new characters and the ink layer is simply cleared. There is no need to save and restore display data thus providing faster throughput.

The S1D13505 hardware cursor/ink layer supports a 2 bpp (four color) overlay image. Two of the available colors are transparent and invert. The remaining two colors are user definable.

7.2 *Registers*

There are a total of eleven registers dedicated to the operation of the hardware cursor/ink layer. Many of the registers need only be set once. Others, such as the positional registers, will be updated frequently.

REG[27h] Ink/Cursor Control Register							
Ink/Cursor Mode Bit 1	Ink/Cursor Mode Bit 0	n/a	n/a	Cursor High Threshold Bit 3	Cursor High Threshold Bit 2	Cursor High Threshold Bit 1	Cursor High Threshold Bit 0

The Ink/Cursor mode bits determine if the hardware will function as a hardware cursor or as an ink layer. See Table 7-1 for an explanation of these bits.

Table 7-1 Ink/Cursor Mode

Register [27h]		Operating Mode
Bit 7	Bit 6	
0	0	Inactive
0	1	Cursor
1	0	Ink
1	1	Reserved

When cursor mode is selected the cursor image is always 64x64 pixels. Selecting an ink layer will result in a large enough area to completely cover the display.

The cursor threshold bits are used to control the Ink/Cursor FIFO depth to sustain uninterrupted display fetches.

REG[28h] Cursor X Position Register 0

Cursor X Position Bit 7	Cursor X Position Bit 6	Cursor X Position Bit 5	Cursor X Position Bit 4	Cursor X Position Bit 3	Cursor X Position Bit 2	Cursor X Position Bit 1	Cursor X Position Bit 0
-------------------------------	-------------------------------	-------------------------------	-------------------------------	-------------------------------	-------------------------------	-------------------------------	-------------------------------

REG[29h] Cursor X Position Register 1

Reserved	n/a	n/a	n/a	n/a	n/a	Cursor X Position Bit 9	Cursor X Position Bit 8
----------	-----	-----	-----	-----	-----	-------------------------------	-------------------------------

Registers [28h] and [29h] control the horizontal position of the hardware cursor. The value in this register specifies the location of the left edge of the cursor. When ink mode is selected these registers should be set to zero.

Cursor X Position bits 9-0 determine the horizontal location of the cursor. With 10 bits of resolution the horizontal cursor range is 1024 pixels.

REG[2Ah] Cursor Y Position Register 0

Cursor Y Position Bit 7	Cursor Y Position Bit 6	Cursor Y Position Bit 5	Cursor Y Position Bit 4	Cursor Y Position Bit 3	Cursor Y Position Bit 2	Cursor Y Position Bit 1	Cursor Y Position Bit 0
-------------------------------	-------------------------------	-------------------------------	-------------------------------	-------------------------------	-------------------------------	-------------------------------	-------------------------------

REG[2Bh] Cursor Y Position Register 1

Reserved	n/a	n/a	n/a	n/a	n/a	Cursor Y Position Bit 9	Cursor Y Position Bit 8
----------	-----	-----	-----	-----	-----	----------------------------	----------------------------

Registers [2Ah] and [2Bh] control the vertical position of the hardware cursor. The value in this register specifies the location of the left edge of the cursor. When ink mode is selected these registers should be set to zero.

Cursor Y Position bits 9-0 determine the location of the cursor. With ten bits of resolution the vertical cursor range is 1024 pixels.

REG[2Ch] Ink/Cursor Color 0 Register 0

Cursor Color 0 Bit 7	Cursor Color 0 Bit 6	Cursor Color 0 Bit 5	Cursor Color 0 Bit 4	Cursor Color 0 Bit 3	Cursor Color 0 Bit 2	Cursor Color 0 Bit 1	Cursor Color 0 Bit 0
-------------------------	-------------------------	-------------------------	-------------------------	-------------------------	-------------------------	-------------------------	-------------------------

REG[2Dh] Ink/Cursor Color 0 Register 1

Cursor Color 0 Bit 15	Cursor Color 0 Bit 14	Cursor Color 0 Bit 13	Cursor Color 0 Bit 12	Cursor Color 0 Bit 11	Cursor Color 0 Bit 10	Cursor Color 0 Bit 9	Cursor Color 0 Bit 8
--------------------------	--------------------------	--------------------------	--------------------------	--------------------------	--------------------------	-------------------------	-------------------------

REG[2Eh] Ink/Cursor Color 1 Register 0

Cursor Color 0 Bit 7	Cursor Color 0 Bit 6	Cursor Color 0 Bit 5	Cursor Color 0 Bit 4	Cursor Color 0 Bit 3	Cursor Color 0 Bit 2	Cursor Color 0 Bit 1	Cursor Color 0 Bit 0
-------------------------	-------------------------	-------------------------	-------------------------	-------------------------	-------------------------	-------------------------	-------------------------

REG[2Fh] Ink/Cursor Color 1 Register 1

Cursor Color 0 Bit 15	Cursor Color 0 Bit 14	Cursor Color 0 Bit 13	Cursor Color 0 Bit 12	Cursor Color 0 Bit 11	Cursor Color 0 Bit 10	Cursor Color 0 Bit 9	Cursor Color 0 Bit 8
--------------------------	--------------------------	--------------------------	--------------------------	--------------------------	--------------------------	-------------------------	-------------------------

Acting in pairs, Registers [2Ch], [2Dh] and registers [2Eh], [2Fh] are used to form the 16 bpp (5-6-5) RGB values for the two user defined colors.

REG[30h] Ink/Cursor Start Address Select Register

Ink/Cursor Start Address Bit 7	Ink/Cursor Start Address Bit 6	Ink/Cursor Start Address Bit 5	Ink/Cursor Start Address Bit 4	Ink/Cursor Start Address Bit 3	Ink/Cursor Start Address Bit 2	Ink/Cursor Start Address Bit 1	Ink/Cursor Start Address Bit 0
--------------------------------------	--------------------------------------	--------------------------------------	--------------------------------------	--------------------------------------	--------------------------------------	--------------------------------------	--------------------------------------

Register [30h] determines the location in the display buffer where the cursor/ink layer will be located. Table 7-2 can be used to determine this location.

Note: Bit 7 is write only, when reading back the register this bit reads a '0'.

Table 7-2 Cursor/Ink Start Address Encoding

Ink/Cursor Start Address Bits [7:0]	Start Address (Bytes)
0	Display Buffer Size - 1024
1 - 0xFF	Display Buffer Size - (n * 8192)

7.3 Limitations

There are limitations for using the hardware cursor/ink layer which should be noted.

REG[29h] and REG[2Bh]

Bit seven of registers [29h] and [2Bh] are write only, and must always be set to zero as setting these bits to one, will cause undefined cursor behavior.

REG[30h]

Bit 7 of register [30h] is write only, therefore programs cannot determine the current cursor/ink layer start address by reading register [30h]. It is suggested that values written to this register be stored elsewhere and used when the current state of this register is required.

No Top/Left Clipping on Hardware Cursor

The S1D13505 does not clip the hardware cursor on the top or left edges of the display. For cursor shapes where the hot spot is not the upper left corner of the image (the hourglass for instance), the cursor image will have to be modified to clip the cursor shape.

7.4 Examples

See Section 12, “Sample Code” for hardware cursor programming examples.

8 *HARDWARE ROTATION*

8.1 *Introduction to Hardware Rotation*

Most computer displays operate in landscape mode. In landscape mode the display is wider than it is high. For instance, a standard display size is 640x480 where the width is 640 pixels and the height is 480 pixels.

Portrait mode rotates the display image clockwise ninety degrees, resulting in a display that is taller than it is wide. Placing the 640x480 display in portrait mode will yield a display that is now 480 pixels wide and 640 pixels high.

8.2 *S1D13505 Hardware Rotation*

The S1D13505 provides hardware support for portrait mode output in 16 and 8 bpp modes.

The switch to portrait mode carries several conditions:

- The (virtual) display offset must be set to 1024 pixels.
- The display start address is calculated differently in portrait mode.
- Calculations that would result in panning in portrait mode result in scrolling in portrait mode and vice-versa.

8.3 Registers

This section will detail each of the registers used to setup portrait mode operations on the S1D13505. The functionality of most of these registers has been covered in previous sections but is included here to make this section complete.

The first step toward setting up portrait mode operation is to set the SwivelView™ Enable bit to 1 (bit 7 of register [0Dh]).

REG[0Dh] Display Mode Register

SwivelView™ Enable	Simultaneous Display Option Select Bit 1	Simultaneous Display Option Select Bit 0	Bit-Per-Pixel Select Bit 2	Bit-Per-Pixel Select Bit 1	Bit-Per-Pixel Select Bit 0	CRT Enable	LCD Enable
--------------------	--	--	----------------------------	----------------------------	----------------------------	------------	------------

Step two involves setting the screen 1 start address registers. Set to 1024 - width for 16 bpp modes and to (1024 - width) / 2 for 8 bpp modes.

REG[10h] Screen 1 Display Start Address Register 0

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
-------	-------	-------	-------	-------	-------	-------	-------

REG[11h] Screen 1 Display Start Address Register 1

Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
--------	--------	--------	--------	--------	--------	-------	-------

REG[12h] Screen 1 Display Start Address Register 2

n/a	n/a	n/a	n/a	Bit 19	Bit 18	Bit 17	Bit 16
-----	-----	-----	-----	--------	--------	--------	--------

Finally set the memory address offset registers to 1024 pixels. In 16 bpp mode load registers [17h:16h] with 1024 and in 8 bpp mode load the registers with 512.

REG[16h] Memory Address Offset Register 0

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
-------	-------	-------	-------	-------	-------	-------	-------

REG[17h] Memory Address Offset Register 1

n/a	n/a	n/a	n/a	n/a	Bit 10	Bit 9	Bit 8
-----	-----	-----	-----	-----	--------	-------	-------

8.4 Limitations

The following limitations apply to SwivelView™:

- Only 8 bpp and 16 bpp modes are supported - 1/2/4 bpp modes are not supported.
- Cursor and ink images are not rotated - software rotation must be used. SwivelView™ must be turned off when the programmer is accessing the Cursor or the ink layer.
- Split screen images appear side-by-side, i.e. the portrait display is split vertically.
- Pixel panning works vertically.

Note: Drawing into the hardware cursor/ink layer with rotation enabled does not work without some form of address manipulation. The easiest way to ensure correct cursor/ink images is to disable SwivelView™, draw in the cursor/ink memory, then re-enable SwivelView™. While writing the cursor/ink memory each pixel must be transformed to its rotated position.

8.5 Examples

Example 7

Enable portrait mode for a 640x480 display at 8 bpp.

Before switching to portrait mode, display memory should be cleared to make the transition smoother. Currently displayed images can not be simply rotated by hardware.

1. The first step toward enabling portrait mode is to set the line offset to 1024 pixels. The Line Offset register is the offset in words.

Write 0x200 to registers [17h]:[16h]. That is write 0x02 to register [17h] and 0x00 to register [16h].

2. The second step to enabling portrait mode is to set the Display 1 Start Address. The Display Start Address registers form a pointer to a word, therefore the value to set the start.

Write 0xC0 (192 or (1024 - 480)/2) to registers [10h], [11h] and [12h]. That is write 0xC) to register [10h], 0x00 to register [11h] and 0x00 to register [12h].

3. Enable display rotation by setting bit 7 of register [0Dh].
4. The display is now configured for portrait mode use. Offset zero into display memory will correspond to the upper left corner of the display. The only difference seen by the programmer will be in acknowledging that the display offset is now 1024 pixels regardless of the physical dimensions of the display surface.

Example 8

Pan the above portrait mode image to the right by 3 pixels then scroll it up by 4 pixels.

Pan the above portrait mode image to the right by 3 pixels then scroll it up by 4 pixels.

1. With portrait mode enabled, the x and y control is rotated as well. Simply swap the x and y co-ordinates and calculate as if the display were not rotated.
2. Calculate the new start address and pixel pan values.

BytesPerScanline = 1024

PixelPan = newX & 0x01;

StartAddr = (newY * BytesPerScanline / 2) + (newX & 0xFFFE) >> 1;

3. Write the start address during the display enabled portion of the frame.
 - a) loop waiting for vertical non-display (b7 of register [0Ah] high).
do register = ReadRegister(0x0A)
while (0x80 != (register & 0x80));
 - b) Loop waiting for the end of vertical non-display.
do register = ReadRegister(0x0A)
while (0x80 == (register & 0x80));
 - c) Write the new start address.
SetRegister(REG_SCRN1_DISP_START_ADDR0, (BYTE) (dwAddr & 0xFF));
SetRegister(REG_SCRN1_DISP_START_ADDR1, (BYTE)((dwAddr >> 8) & 0xFF));
SetRegister(REG_SCRN1_DISP_START_ADDR2, (BYTE)((dwAddr >> 16) & 0x0F));
do register = ReadRegister(0x0A)
while (0x80 == (register & 0x80));
4. Write the pixel pan value during the vertical non-display portion of the frame.
 - a) Coming from the above code wait for beginning of the non-display period.
do register = ReadRegister(0x0A)
while (0x80 != (register & 0x80));
 - b) Write the new pixel panning value.
register = ReadRegister(0x18);
register &= 0xF0;
register |= (PixelPan & 0x0F);
WriteRegister(0x18, register);

9 *CRT CONSIDERATIONS*

9.1 *Introduction*

The S1D13505 is capable of driving either an LCD panel, or a CRT display, or both simultaneously.

As display devices, panels tend to be lax in their horizontal and vertical timing requirements. CRT displays often cannot vary by more than a very small percentage in their timing requirements before the image is degraded.

Central to the following sections are VESA timings. Rather than fill this section of the guide with pages full of register values it is recommended. For more information on VESA timings contact the Video Electronics Standards association.

CRT Only

All CRT output should meet VESA timing specifications. The VESA specification details all the parameters of the display and non-display times as well as the input clock required to meet the times.

Simultaneous Display

As mentioned in the previous section, CRT timings should always comply to the VESA specification. This requirement implies that during simultaneous operation the timing must still be VESA compliant. For most panels, being run at CRT frequencies is not a problem. One side effect of running with these usually slower timings will be a flicker on the panel.

One limitation of simultaneous display is that should a dual panel be the second display device the half frame buffer must be disabled for correct operation.

10 IDENTIFYING THE S1D13505

Identification of the S1D13505 can only occur after the host interface has been enabled. From reset the steps to identifying the S1D13505 are as follows:

1. If hardware suspend has been enabled then disable the suspend. On the S1D13505 ISA evaluation board this is accomplished by performing a read operation to address 0xF00000.
2. Write a 00h to register [1B] to enable the host interface.
3. Read register [00h]
4. The production version of the S1D13505 is 0x0C.

11 *HARDWARE ABSTRACTION LAYER (HAL)*

11.1 *Introduction*

The HAL is a processor independent programming library provided by Seiko Epson. The HAL was developed to aid the implementation of internal test programs. The HAL provides an easy, consistent method of programming the S1D13505 on different processor platforms. The HAL also allows for easier porting of programs between S1D1350x products.

The HAL keeps sample code simpler, although end programmers may find the HAL functions to be limited in their scope, and may wish to ignore the HAL.

11.2 *API for 13505HAL*

The following is a description of the HAL library. Updates and revisions to the HAL may include new functions not included in the following documentation

Initialization

The following section describes the HAL functions dealing with initialization of the S1D13505. Typically a programmer has only to concern themselves with calls to `seRegisterDevice()` and `seSetInit()`.

int seRegisterDevice(const LPHAL_STRUC lpHalInfo, int * pDevice)

Description: Register the S1D13505 device parameters with the HAL library. The device parameters have been configured with address range, register values, desired frame rate, etc., and have been saved in the HAL_STRUCT structure pointed to by lpHalInfo. Additionally this routine allocates, from system memory, address space for accessing registers and the display buffer.

Parameters: lpHalInfo - pointer to HAL_STRUCT information structure
pDevice - pointer to the integer to receive the device ID

Return Value: ERR_OK - operation completed with no problems

Note: No S1D13505 registers are changed by calling seRegisterDevice().

int seSetInit(int DevID)

Description: Configures the system for operation using the specified default settings. Everything required for operation is set in this call.

Parameters: DevID - registered device ID

Return Value: ERR_OK - operation completed with no problems
ERR_FAILED - unable to complete operation. Occurs as a result of requesting parameters that exceed timing specifications.

Notes: • This function and seSetDisplayMode() are nearly identical. The largest difference is that seSetInit() always uses the configuration designated to be the default by 13505CFG.EXE and seSetDisplayMode() allows the programmer to select which configuration.

- This routine does not configure the Look-Up Tables.

int seSetDisplayMode(int DevID, int DisplayMode, int flags)

Description: This routine sets the S1D13505 registers according to the values contained in the HAL_STRUCT register section referred to by DisplayMode.

Setting all the registers means that timing, display surface dimensions,... all aspects of chip operation are set with this call.

Parameters: DevID - a valid registered device ID
DisplayMode - the HAL_STRUCT register set to use:
DISP_MODE_LCD,
DISP_MODE_CRT, or
DISP_MODE_SIMULTANEOUS
flags - can be set to one or more flags. Each flag added by using the logical OR command. Do not add mutually exclusive flags. Flags can be set to 0 to use defaults.
DONT_CLEAR_MEM (default) - do not clear memory
CLEAR_MEM - clear display buffer memory
DISP_FIFO_OFF - turn off display FIFO
(blank screen except for cursor or ink layer)
DISP_FIFO_ON (default) - turn on display FIFO

Return Value: ERR_OK - no problems encountered
ERR_FAILED - unable to complete operation. Occurs as a result of requesting parameters that exceed timing specifications.

Example: seSetDisplayMode(DevID, DISP_MODE_LCD, CLEAR_MEM | DISP_FIFO_OFF);
The above will initialize for the LCD, and then clear display buffer memory and blank the screen. The advantage to this approach is that afterwards the application can write to the display without showing the image until memory is completely updated; the application would then call seDisplayFIFO(DevID, ON).

int seInitHal(void)

Description: This function initializes variable used by the HAL library. This function must be called once when the application starts.

Normally programmers do not have to concern themselves with seInitHal(). On PC platforms, seRegisterDevice() automatically calls seInitHal(). Consecutive calls to seRegisterDevice() will not call seInitHal() again. On non-PC platforms the start-up code, supplied by Seiko, will call seInitHal(). If however support code for a new operating platform is written the programmer must ensure that seInitHAL is called prior to calling other HAL functions.

Parameters: None

Return Value: ERR_OK - operation completed with no problems

General HAL Support

General HAL support covers the miscellaneous functions. There is usually no more than one or two functions devoted to any particular aspect of S1D13505 operation.

int seGetId(int DevID, int * pId)

Description: Reads the S1D13505 revision code register to determine the chip product and revisions. The interpreted value is returned in pId.

Parameters: Device - registered device ID
pId - pointer to the byte to receive the controller ID.

For the S1D13505 the return values are currently:

ID_S1D13505

ID_S1D13505F00A

ID_UNKNOWN

Other HAL libraries will return their respective controller IDs upon detection of their controller.

Return Value: ERR_OK - operation completed with no problems
ERR_UNKNOWN_DEVICE - returned when pId returns ID_UNKNOWN. The HAL was unable to identify the display controller.

Note: seGetId() will disable hardware suspend Intel platforms, and will enable the host interface (register [1Bh]) on all platforms.

void seGetHalVersion(const char ** pVersion, const char ** pStatus, const char **pStatusRevision)

Description: Retrieves the HAL library version. The return values are all ASCII strings. A typical return would be: "1.01" (HAL version 1.01), "B" (The 'B' is the beta designator), "5" (This example would be Beta5, if pStatus is NULL the so should pStatusRevision).

Parameters: pVersion - must point to an allocated string of size VER_SIZE
pStatus - must point to an allocated string of size STATUS_SIZE
pStatusRevision - must point to an allocated string of size STAT_REV_SIZE

Return Value: None

int seGetMemSize(int DevID, DWORD * pSize)

Description: This routine returns the amount of installed video memory. The memory size is determined by reading the status of MD6 and MD7. *pSize will be set to either 0x80000 (512 KB) or 0x200000 (2 MB).

Parameters: DevID - registered device ID
pSize - pointer to a DWORD to receive the size

Return Value: ERR_OK - the operation completed successfully

int seGetLastUsableByte(int DevID, DWORD * pLastByte)

Description: Calculates the offset of the last byte in the display buffer which can be used by applications. Locations following LastByte are reserved for system use. Items such as the half frame buffer, hardware cursor and ink layer will be located in memory from GetLastUsableByte() + 1 to the end of memory.

It is assumed that the registers will have been initialized before calling seGetLastUsableByte(). Factors such as the half frame buffer and hardware cursor / ink layer being enabled dynamically alter the amount of display buffer available to an application. Call seGetLastUsableByte() any time the true end of usable memory is required.

Parameters: DevID - registered device ID
pLastByte - pointer to a DWORD to receive the offset to the last usable byte of display buffer

Return Value: ERR_OK - operation completed with no problems

int seGetBytesPerScanline(int DevID, UINT * pBytes)

Description: Determines the number of bytes per scan line of current display mode. It is assumed that the registers have already been correctly initialized before seGetBytesPerScanline() is called.

The number of bytes per scanline calculation includes the value in the offset register. For rotated modes the return value will be either 1024 or 2048 to reflect the 1024 x 1024 virtual area of the rotated memory.

Parameters: DevID - registered device ID
pBytes - pointer to an integer which indicates the number of bytes per scan line

Return Value: ERR_OK - operation completed with no problems
ERR_FAILED - returned when this function is called for rotated display modes other than 8 or 16 bpp.

int seGetScreenSize(int DevID, UINT * Width, UINT * Height)

Description: Gets the width and height in pixels of the display surface. The width and height are derived by reading the horizontal and vertical timing registers and calculating the dimensions.

When the display is in portrait mode the dimensions will be swapped. (i.e. a 640x480 display in portrait mode will return a width and height of 480 and 640, respectively).

Parameters: DevID - registered device ID
Width - unsigned integer to receive the display width
Height - unsigned integer to receive the display height

Return value: ERR_OK - the operation completed successfully

int seSelectBusWidth(int DevID, int Width)

- Description:** Call this function to select the interface bus width on the ISA evaluation card. Selectable widths are 8 bit and 16 bit.
- Parameters:** DevID - registered device ID
Width - desired bus width. Must be 8 or 16.
- Return Value:** ERR_OK - the operation completed successfully
ERR_FAILED - the function was called on a non-ISA platform or width was not set to 8 or 16.

Note: This call applies to the S1D13505 ISA evaluation cards only.

int seGetHostBusWidth(int DevID, int * Width)

- Description:** This function retrieves the default (as set by 13505CFG.EXE) value for the host bus interface width and returns it in Width.
- Parameters:** DevID - registered device ID
Width - integer to hold the returned value of the host bus width
- Return Value:** ERR_OK - the function completed successfully

int seDisplayEnable(int DevID, BYTE State)

- Description:** This routine turns the display on or off by enabling or disabling the ENABLE bit of the display device (PANEL, CRT, or SIMULTANEOUS). The configuration defined in 13505CFG determines which device(s) will be affected.
- Parameters:** DevID - registered device ID
State - set to ON or OFF to respectively enable or disable the display
- Return Value:** ERR_OK - the function completed successfully

int seDisplayFifo(int DevID, BYTE State)

- Description:** This routine turns the display on or off by enabling or disabling the display FIFO (the hardware cursor and ink layer are not affected).
- Enabling and disabling the display FIFO has a much faster and cleaner appearing effect when the display is to be blanked and it allows full CPU bandwidth to the display buffer.
- Parameters:** DevID - registered device ID
State - set to ON or OFF to respectively enable or disable the display FIFO
- Return Value:** ERR_OK - the function completed successfully

Note: Disabling the display FIFO will force all display data outputs to zero but horizontal and vertical sync pulses and panel power supply are still active.

int seDelay(int DevID, DWORD Seconds)

Description: This function will delay for the number of seconds given in *Seconds* before returning to the caller.

This function was originally intended for non-PC platforms. Information on how to access the timers was not always immediately available however we do know frame rate and can use that for timing calculations. The S1D13505 registers must be initialized for this function to work correctly.

PC platform function calls the C timing functions and is therefore independent of the register settings.

Parameters: DevID - registered device ID
Seconds - time to delay in seconds

Return Value: ERR_OK - operation completed with no problems
ERR_FAILED - returned only on non-PC platforms when the S1D13505 registers have not been initialized

Advanced HAL Functions

Advanced HAL functions include the functions to support split and virtual screen operation and are the same features that were described in the section on advanced programming techniques.

int seSplitInit(int DevID, DWORD Scrn1Addr, DWORD Scrn2Addr)

Description: This function prepares the system for split screen operation. In order for split screen to function the starting address in display buffer for the upper portion - screen 1, and the lower portion - screen 2 must be specified. Screen 1 is always displayed above screen 2 on the display regardless of the location of their respective starting addresses.

Parameters: DevID - registered device ID
Scrn1Addr - offset in display buffer, in bytes, to the start of screen 1
Scrn2Addr - offset in display buffer, in bytes, to the start of screen 2

Return Value: ERR_OK - operation completed with no problems

Note: It is assumed that the system has been properly initialized prior to calling seSplitInit().

int seSplitScreen(int DevID, int WhichScreen, long VisibleScanlines)

Description: Changes the relevant registers to adjust the split screen according to the number of visible lines requested. *WhichScreen* determines which screen, screen 1 or screen 2, to change.

The smallest screen 1 can be set to is one line. This is due to the way the register values are used internally on the S1D13505. Setting the line compare register to zero results in one line of screen 1 being displayed before starting on screen 2.

Parameters: DevID - registered device ID
WhichScreen - must be set to 1 or 2, or use the constants SCREEN1 or SCREEN2, to identify which screen to base calculations on
VisibleScanlines - number of lines to show for the selected screen

Return Value: ERR_OK - operation completed with no problems
ERR_HAL_BAD_ARG - argument VisibleScanlines is negative or is greater than vertical panel size or WhichScreen is not SCREEN1 or SCREEN2.

Note: seSplitInit() must be called before calling seSplitScreen().

int seVirtInit(int DevID, DWORD VirtX, DWORD * VirtY)

- Description:** This function prepares the system for virtual screen operation. The programmer passes the desired virtual width, in pixels, as *VirtX*. When the routine returns *VirtY* will contain the maximum number of line that can be displayed at the requested virtual width.
- Parameter:**
- | | |
|-------|---|
| DevID | - registered device ID |
| VirtX | - horizontal size of virtual display in pixels.
(Must be greater or equal to physical size of display) |
| VirtY | - a return placeholder for the maximum number of lines available at the requested and returns value in yVirt. |
- Return Value:**
- | | |
|-----------------|--|
| ERR_OK | - operation completed with no problems |
| ERR_HAL_BAD_ARG | - returned in three situations |
| | 1) the virtual width (VirtX) is greater than the largest attainable width
(The maximum allowable xVirt is $0x3FF * (16 / \text{bpp})$) |
| | 2) the virtual width is less than the physical width or |
| | 3) the maximum number of lines is less than the physical number of lines |

Note: The system must have been properly initialized prior to calling seVirtInit().

int seVirtMove(int DevID, int WhichScreen, DWORD x, DWORD y)

- Description:** This routine pans and scrolls the display. In the case where split screen operation is being used the WhichScreen argument specifies which screen to move. The x and y parameters specify, in pixels, the starting location in the virtual image for the top left corner of the applicable display.
- Parameter:**
- | | |
|-------------|--|
| DevID | - registered device ID |
| WhichScreen | - must be set to 1 or 2, or use the constants SCREEN1 or SCREEN2, to identify which screen to base calculations on |
| x | - new starting X position in pixels |
| y | - new starting Y position in pixels |
- Return Value:**
- | | |
|-----------------|---|
| ERR_OK | - operation completed with no problems |
| ERR_HAL_BAD_ARG | - there are several reasons for this return value: |
| | 1) WhichScreen is not SCREEN1 or SCREEN2. |
| | 2) the y argument is greater than the last available line less the screen height. |

Note: seVirtInit() must be been called before calling seVirtMove().

Register / Memory Access

The Register/Memory Access functions provide access to the S1D13505 registers and display buffer through the HAL.

int seSetReg(int DevID, int Index, BYTE Value)

Description: Writes Value to the register specified by Index.

Parameters: DevID - registered device ID
 Index - register index to set
 Value - value to write to the register

Return Value: ERR_OK - operation completed with no problems

int seGetReg(int DevID, int Index, BYTE * pValue)

Description: Reads the value in the register specified by index.

Parameters: Device - registered device ID
 Index - register index to read
 pValue - return value of the register

Return Value: ERR_OK - operation completed with no problems

int seWriteDisplayBytes(int DevID, DWORD Offset, BYTE Value, DWORD Count)

Description: This routine writes one or more bytes to display buffer at the offset specified by Addr. If a count greater than one is specified all bytes will have the same value.

Parameters: DevID - registered device ID
 Offset - offset from start of the display buffer
 Value - BYTE value to write
 Count - number of bytes to write

Return Value: ERR_OK - operation completed with no problems
 ERR_HAL_BAD_ARG - if the value for Addr is greater than the amount of installed memory or if Addr plus Count is greater than the installed memory.

int seWriteDisplayWords(int DevID, DWORD Offset, WORD Value, DWORD Count)

Description: Writes one or more words to the display buffer.

Parameters: DevID - registered device ID
 Offset - offset from start of the display buffer
 Value - WORD value to write
 Count - number of words to write

Return Value: ERR_OK - operation completed with no problems
 ERR_HAL_BAD_ARG - if the value for Addr is greater than the amount of installed memory or if Addr plus Count is greater than the installed memory.

int seWriteDisplayDwords(int DevID, DWORD Offset, DWORD Value, DWORD Count)

Description: Writes one or more dwords to the display buffer.

Parameters: DevID - registered device ID
Offset - offset from start of the display buffer
Value - DWORD value to write
Count - number of dwords to write

Return Value: ERR_OK - operation completed with no problems
ERR_HAL_BAD_ARG - if the value for Addr is greater than the amount of installed memory or if Addr plus Count is greater than the installed memory.

int seReadDisplayByte(int DevID, DWORD Offset, BYTE *pByte)

Description: Reads a byte from the display buffer at the specified offset and returns the value in pByte.

Parameters: DevID - registered device ID
Offset - offset, in bytes, from start of the display buffer
pByte - return value of the display buffer location.

Return Value: ERR_OK - operation completed with no problems
ERR_HAL_BAD_ARG - if the value for Addr is greater than the amount of installed memory.

int seReadDisplayWord(int DevID, DWORD Offset, WORD *pWord)

Description: Reads a word from the display buffer at the specified offset and returns the value in pWord.

Parameters: DevID - registered device ID
Offset - offset, in bytes, from start of the display buffer
pWord - return value of the display buffer location

Return Value: ERR_OK - operation completed with no problems
ERR_HAL_BAD_ARG - if the value for Addr is greater than the amount of installed memory.

int seReadDisplayDword(int DevID, DWORD Offset, DWORD *pDword)

Description: Reads a dword from the display buffer at the specified offset and returns the value in pDword.

Parameters: DevID - registered device ID
Offset - offset from start of the display buffer
pDword - return value of the display buffer location

Return Value: ERR_OK - operation completed with no problems
ERR_HAL_BAD_ARG - if the value for Addr is greater than the amount of installed memory.

Color Manipulation

The functions in the Color Manipulation section deal with altering the color values in the Look-Up Table directly through the accessor functions and indirectly through the color depth setting functions.

int seSetLut(int DevID, BYTE *pLut, int Count)

Description: This routine can write one or more LUT entries. The writes always start with Look-Up Table index 0 and continue for *Count* entries.

A Look-Up Table entry consists of three bytes, one each for Red, Green, and Blue. The color information is stored in the four most significant bits of each byte.

Parameters:

- DevID - registered device ID
- pLut - pointer to an array of BYTE lut[16][3]
- lut[x][0] == RED component
- lut[x][1] == GREEN component
- lut[x][2] == BLUE component
- Count - the number of LUT entries to write.

Return Value: ERR_OK - operation completed with no problems

int seGetLut(int DevID, BYTE *pLUT, int Count)

Description: This routine reads one or more LUT entries and puts the result in the byte array pointed to by pLUT.

A Look-Up Table entry consists of three bytes, one each for Red, Green, and Blue. The color information is stored in the four most significant bits of each byte.

Parameters:

- DevID - registered device ID
- pLUT - pointer to an array of BYTE lut[16][3]
pLUT must point to enough memory to hold *Count* x 3 bytes of data.
- Count - the number of LUT elements to read.

Return Value: ERR_OK - operation completed with no problems

int seSetLutEntry(int DevID, int Index, BYTE *pEntry)

Description: This routine writes one LUT entry. Unlike seSetLut, the LUT entry indicated by *Index* can be any value from 0 to 255.

A Look-Up Table entry consists of three bytes, one each for Red, Green, and Blue. The color information is stored in the four most significant bits of each byte.

Parameters:

- DevID - registered device ID
- Index - index to LUT entry (0 to 15)
- pLUT - pointer to an array of three bytes.

Return Value: ERR_OK - operation completed with no problems

int seGetLutEntry(int DevID, int index, BYTE *pEntry)

Description: This routine reads one LUT entry from any index.

Parameters: DevID - registered device ID
Index - index to LUT entry (0 to 15)
pEntry - pointer to an array of three bytes

Return Value: ERR_OK - operation completed with no problems

int seSetBitsPerPixel(int DevID, UINT BitsPerPixel)

Description: This routine sets the system color depth. Valid arguments for *BitsPerPixel* is are: 1, 2, 4, 8, 15, and 16.

After performing validity checks for the requested color depth the appropriate registers are changed and the Look-Up Table is set its default value.

This call is similar to a mode set call on a standard VGA.

Parameter: DevID - registered device ID
BitsPerPixel - desired color depth in bits per pixel

Return Value: ERR_OK - operation completed with no problems
ERR_FAILED - possible causes for this error message include:

- 1) attempted to set 15/16 bpp at 800x600 resolution (not supported on the S1D13505)
- 2) attempted to set other than 8 or 15/16 bpp in portrait mode (portrait mode only supports 8 and 15/16 bpp)
- 3) factors such as input clock and memory speed will affect the ability to set some color depths. If the requested color depth cannot be set this call will fail

int seGetBitsPerPixel(int DevID, UINT * pBitsPerPixel)

Description: This function reads the S1D13505 registers to determine the current color depth and returns the result in *pBitsPerPixel*.

Determines the color depth of current display mode.

Parameters: DevID - registered device ID
pBitsPerPixel - return value is the current color depth

Return Value: ERR_OK - operation completed with no problems

Drawing

The Drawing section covers HAL functions that deal with displaying pixels, lines and shapes.

int seSetPixel(int DevID, long x, long y, DWORD Color)

Description: Draws a pixel at coordinates x,y in the requested color. This routine can be used for any color depth.

Parameters:

- DevID - registered device ID
- x - horizontal coordinate of the pixel (starting from 0)
- y - vertical coordinate of the pixel (starting from 0)
- Color - at 1, 2, 4, and 8 bpp Color is an index into the LUT. At 15 and 16 bpp Color defines the color directly (i.e. rrrrrgggggbbbb)

Return Value: ERR_OK - operation completed with no problems

int seGetPixel(int DevID, long x, long y, DWORD *pColor)

Description: Reads the pixel color at coordinates x,y. This routine can be used for any color depth.

Parameters:

- DevID - registered device ID
- x - horizontal coordinate of the pixel (starting from 0)
- y - vertical coordinate of the pixel (starting from 0)
- pColor - at 1, 2, 4, and 8 bpp pColor points to an index into the LUT. At 15 and 16 bpp pColor points to the color directly (i.e. rrrrrgggggbbbb)

Return Value: ERR_OK - operation completed with no problems

int seDrawLine(int DevID, long x1, long y1, long x2, long y2, DWORD Color)

Description: This routine draws a line on the display from the endpoints defined by x1,y1 to x2,y2 in the requested Color.

Currently seDrawLine() only draws horizontal and vertical lines.

Parameters:

- Device - registered device ID
- (x1, y1) - top left corner of line
- (x2, y2) - bottom right corner of line (see note below)
- Color - color of line

- for 1, 2, 4, and 8 bpp, 'Color' refers to the pixel value which points to the respective LUT/DAC entry.

- for 15 and 16 bpp, 'Color' refers to the pixel value which stores the red, green, and blue intensities within a WORD.

Return Value: ERR_OK - operation completed with no problems
ERR_INVALID_REG_DEVICE - device argument is not valid.

int seDrawRect(int DevID, long x1, long y1, long x2, long y2, DWORD Color, BOOL SolidFill)

- Description:** This routine draws and optionally fills a rectangular area of display buffer. The upper right corner of the rectangle is defined by x1,y1 and the lower right corner is defined by x2,y2. The color, defined by *Color*, applies to the border and to the optional fill.
- Parameters:**
- DevID - registered device ID
 - x1, y1 - top left corner of the rectangle (in pixels)
 - x2, y2 - bottom right corner of the rectangle (in pixels)
 - Color - the color to draw the rectangle outline and fill with
 - at 1, 2, 4, and 8 bpp Color is an index into the Look-Up Table.
 - at 15/16 bpp Color defines the color directly (i.e. rrrrrgggggbbbb)
 - SolidFill - flag whether to fill the rectangle or simply draw the border.
 - set to 0 for no fill, set to non-0 to fill the inside of the rectangle
- Return Value:** ERR_OK - operation completed with no problems

int seDrawEllipse(int DevID, long xc, long yc, long xr, long yr, DWORD Color, BOOL SolidFill)

- Description:** This routine draws an ellipse with the center located at xc,yc. The xr and yr parameters specify the x and y radii, in pixels, respectively. The ellipse will be drawn in the color specified in 'Color'.
- Parameters:**
- DevID - registered device ID
 - xc, yc - the center location of the ellipse (in pixels)
 - xr - horizontal radius of the ellipse (in pixels)
 - yr - vertical radius of the ellipse (in pixels)
 - Color - the color to draw the ellipse
 - at 1, 2, 4, and 8 bpp Color is an index into the Look-Up Table.
 - at 15/16 bpp Color defines the color directly (i.e. rrrrrgggggbbbb)
 - SolidFill - unused
- Return Value:** ERR_OK - operation completed with no problems
- Note:** The 'SolidFill' argument is currently unused and is included for future considerations.

int seDrawCircle(int DevID, long xc, long yc, long Radius, DWORD Color, BOOL SolidFill)

- Description:** This routine draws a circle with the center located at xc,yc and a radius of Radius. The circle will be drawn in the color specified in *Color*.
- Parameters:**
- DevID - registered device ID
 - xc, yc - the center of the circle (in pixels)
 - Radius - the circle's radius (in pixels)
 - Color - the color to draw the ellipse
 - at 1, 2, 4, and 8 bpp Color is an index into the Look-Up Table.
 - at 15/16 bpp Color defines the color directly (i.e. rrrrrgggggbbbb)
 - SolidFill - unused
- Return Value:** ERR_OK - operation completed with no problems
- Note:** The *SolidFill* argument is currently unused and is included for future considerations.

Hardware Cursor

The routines in this section support hardware cursor functionality. Several of the calls look similar to normal drawing calls (i.e. `seDrawCursorLine()`) however these calls remove the programmer from having to know the particulars of the cursor memory location, layout and whether portrait mode is enabled.

int seInitCursor(int DevID)

Description: Prepares the hardware cursor for use. This consists of determining a location in display buffer for the cursor, setting cursor memory to the transparent color and enabling the cursor.

When this call returns the cursor is enabled, the cursor image is transparent and ready to be drawn.

Parameters: DevID - a registered device ID

Return Value: ERR_OK - operation completed with no problems

int seCursorOn(int DevID)

Description: This function enables the cursor after it has been disabled through a call to `seCursorOff()`. After enabling the cursor will have the same shape and position as it did prior to being disabled. The exception to the size and position occurs if the ink layer was used while the cursor was disabled.

Parameters: DevID - a registered device ID

Return Value: ERR_OK - operation completed with no problems

int seCursorOff(int DevID)

Description: This routine disables the cursor. While disabled the cursor is invisible.

Parameters: DevID - a registered device ID

Return Value: ERR_OK - operation completed with no problems

int seGetCursorStartAddr(int DevID, DWORD * Offset)

Description: This function retrieves the offset to the first byte of hardware cursor memory.

Parameters: DevID - a registered device ID
Offset - a DWORD to hold the return value.

Return Value: ERR_OK - operation completed with no problems

int seMoveCursor(int DevID, long x, long y)

Description: Moves the upper left corner of the hardware cursor to the pixel position x,y.

Parameters: DevID - a registered device ID
x, y - the x,y position (in pixels) to move the cursor to

Return Value: ERR_OK - operation completed with no problems

int seSetCursorColor(int DevID, int Index, DWORD Color)

Description: Sets the color of the specified cursor index to 'Color'. The user definable hardware cursor colors are 16-bit 5-6-5 RGB colors.

The hardware cursor image is always 2 bpp or four colors. Two of the colors are defined to be transparent and inverse. This leaves two colors which are user definable.

Parameters: DevID - a registered device ID
Index - the cursor index to set. Valid values are 0 and 1
Color - a DWORD value which hold the requested color

Return Value: ERR_OK - operation completed with no problems
ERR_FAILED - returned if Index if other than 0 or 1

int seSetCursorPixel(int DevID, long x, long y, DWORD Color)

Description: Draws a single pixel into the hardware cursor. The pixel will be of color 'Color' located at x,y pixels relative to the top left of the hardware cursor.

The value of 'Color' must be 0 to 3. Values 0 and 1 refer to the two user definable colors. If 'Color' is 2 then the pixel will be transparent and if the value is 3 the pixel will be an inversion of the underlying screen color.

Parameters: DevID - a registered device ID
x, y - draw coordinates, in pixels, relative to the top left corner of the cursor
Color - a value of 0 to 3 to draw the pixel with

Return Value: ERR_OK - operation completed with no problems

int seDrawCursorLine(int DevID, long x1, long y1, long x2, long y2, DWORD Color)

Description: Draws a line between the two endpoints, x1,y1 and x2,y2, in the hardware cursor display buffer using color 'Color'.

The value of 'Color' must be 0 to 3. Values 0 and 1 refer to the two user definable colors. If 'Color' is 2 then the pixel will be transparent and if the value is 3 the pixel will be an inversion of the underlying screen color.

Parameters: DevID - a registered device ID
x1,y1 - first line endpoint (in pixels)
x2,y2 - second line endpoint (in pixels)
Color - a value of 0 to 3 to draw the pixel with

Return Value: ERR_OK - operation completed with no problems

int seDrawCursorRect(int DevID, long x1, long y1, long x2, long y2, DWORD Color, BOOL SolidFill)

Description: This routine will draw a rectangle in hardware cursor memory. The upper left corner of the rectangle is defined by the point x1,y1 and the lower right is the point x2,y2. Both points are relative to the upper left corner of the cursor.

The value of 'Color' must be 0 to 3. Values 0 and 1 refer to the two user definable colors. If 'Color' is 2 then the pixel will be transparent and if the value is 3 the pixel result will be an inversion of the underlying screen color.

If 'SolidFill' is specified the interior of the rectangle will be filled with 'Color', otherwise the rectangle is only outlined in 'Color'.

Parameters:

DevID	- a registered device ID
x1,y1	- upper left corner of the rectangle (in pixels)
x2,y2	- lower right corner of the rectangle (in pixels)
Color	- a 0 to 3 value to draw the rectangle with
SolidFill	- flag for filling the rectangle interior
	- if equal to 0 then outline the rectangle if not equal to 0 then fill the rectangle

Return Value: ERR_OK - operation completed with no problems

int seDrawCursorEllipse(int DevID, long xc, long yc, long xr, long yr, DWORD Color, BOOL SolidFill)

Description: This routine draws an ellipse within the hardware cursor display buffer. The ellipse will be centered on the point xc,yc and will have a horizontal radius of xr and a vertical radius of yr.

The value of 'Color' must be 0 to 3. Values 0 and 1 refer to the two user definable colors. If 'Color' is 2 then the pixel will be transparent and if the value is 3 the pixel will be an inversion of the underlying screen color.

Currently seDrawCursorEllipse() does not support solid fill of the ellipse.

Parameters:

DevID	- a registered device ID
xc, yc	- center of the ellipse (in pixels)
xr	- horizontal radius (in pixels)
yr	- vertical radius (in pixels)
Color	- 0 to 3 value to draw the pixels with
SolidFill	- flag to solid fill the ellipse (not currently used)

Return Value: ERR_OK - operation completed with no problems

int seDrawCursorCircle(int DevID, long x, long y, long Radius, DWORD Color, BOOL SolidFill)

Description: This routine draws a circle in hardware cursor display buffer. The center of the circle will be at x,y and the circle will have a radius of 'Radius' pixels.

The value of 'Color' must be 0 to 3. Values 0 and 1 refer to the two user definable colors. If 'Color' is 2 then the pixel will be transparent and if the value is 3 the pixel will be an inversion of the underlying screen color.

Currently seDrawCursorCircle() does not support the solid fill option.

Parameters: DevID - a registered device ID
 x,y - center of the circle (in pixels)
 Radius - radius of the circle (in pixels)
 Color - 0 to 3 value to draw the circle with
 SolidFill - flag to solid fill the circle (currently not used)

Return Value: ERR_OK - operation completed with no problems

Ink Layer

The functions in this section support the hardware ink layer. Overall these functions are nearly identical to the hardware cursor routines. In fact the same S1D13505 hardware is used for both features which means that only the cursor or the ink layer can be active at any given time. The difference between the hardware cursor and the ink layer is that in cursor mode the image is a maximum of 64x64 pixels and can be moved around the display while in ink layer mode the image is as large as the display and is in a fixed position. In both cases the number of colors and the way the colors are handled are identical.

int selnitlnk(int DevID)

Description: This routine prepares the ink layer for use. This consists of determining the start address for the ink layer, setting the ink layer to the transparent color and enabling the ink layer.

When this function returns the ink layer is enable, transparent and ready to be drawn on.

Parameters: DevID - a registered device ID

Return Value: ERR_OK - operation completed with no problems
 ERR_FAILED - if the ink layer cannot be enabled do to timing constraints this value will be returned.

int selnkon(int DevID)

Description: Enables the ink layer after a call to seInkOff(). If the hardware cursor has not been used between the time seInkOff() was called and this call then the contents of the ink layer should be exactly as it was prior to the call to seInkOff().

Parameters: DevID - a registered device ID

Return Value: ERR_OK - operation completed with no problems

int seInkOff(int DevID)

Description: Disables the ink layer. When disabled the ink layer is not visible.

Parameters: DevID - a registered device ID

Return Value: ERR_OK - operation completed with no problems

int seGetInkStartAddr(int DevID, DWORD * Offset)

Description: This function retrieves the offset to the first byte of hardware ink layer memory.

Parameters: DevID - a registered device ID
Offset - a DWORD to hold the return value.

Return Value: ERR_OK - operation completed with no problems

int seSetInkColor(int DevID, int Index, DWORD Color)

Description: Sets the color of the specified cursor index to 'Color'. The user definable hardware cursor colors are sixteen bit 5-6-5 RGB colors.

The hardware cursor image is always 2 bpp or four colors. Two of the colors are defined to be transparent and inverse. This leaves two colors which are user definable.

Parameters: DevID - a registered device ID
Index - the index, 0 or 1, to write the color to
Color - a sixteen bit RRRRRGGGGGBBBB color to write to 'Index'

Return Value: ERR_OK - operation completed with no problems
ERR_FAILED - an index other than 0 or 1 was specified.

int seSetInkPixel(int DevID, long x, long y, DWORD Color)

Description: Sets one pixel located at x,y to the value 'Color'. The point x,y is relative to the upper left corner of the display.

The value of 'Color' must be 0 to 3. Values 0 and 1 refer to the two user definable colors. If 'Color' is 2 then the pixel will be transparent and if the value is 3 the pixel will be an inversion of the underlying screen color.

Parameters: DevID - a registered device ID
x,y - coordinates of the pixel to draw
Color - a 0 to 3 value to draw the pixel with

Return Value: ERR_OK - operation completed with no problems

int seDrawInkLine(int DevID, long x1, long y1, long x2, long y2, DWORD Color)

Description: This routine draws a line in 'Color' between the endpoints x1,y1 and x2,y2.

The value of 'Color' must be 0 to 3. Values 0 and 1 refer to the two user definable colors. If 'Color' is 2 then the pixel will be transparent and if the value is 3 the pixel will be an inversion of the underlying screen color.

Parameters:

DevID	- a registered device ID
x1,y1	- first endpoint of the line (in pixels)
x1,y2	- second endpoint of the line (in pixels)
Color	- a value from 0 to 3 to draw the line with

Return Value: ERR_OK - operation completed with no problems

int seDrawInkRect(int DevID, long x1, long y1, long x2, long y2, DWORD Color, BOOL SolidFill)

Description: Draws a rectangle of color 'Color' and optionally fills it. The upper left corner of the rectangle is the point x1,y1 and the lower right corner of the rectangle is the point x2,y2.

The value of 'Color' must be 0 to 3. Values 0 and 1 refer to the two user definable colors. If 'Color' is 2 then the pixel will be transparent and if the value is 3 the pixel will be an inversion of the underlying screen color.

Parameters:

DevID	- a registered device ID
x1,y1	- upper left corner of the rectangle (in pixels)
x2,y2	- lower right corner of the rectangle (in pixels)
Color	- a two bit value (0 to 3) to draw the rectangle with
SolidFill	- a flag to indicate the interior should be filled

Return Value: ERR_OK - operation completed with no problems

int seDrawInkEllipse(int DevID, long xc, long yc, long xr, long yr, DWORD Color, BOOL SolidFill)

Description: This routine draws an ellipse with the center located at xc,yc. The xr and yr parameters specify the x and y radii, in pixels, respectively. The ellipse will be drawn in the color specified by 'Color'.

The value of 'Color' must be 0 to 3. Values 0 and 1 refer to the two user definable colors. If 'Color' is 2 then the pixel will be transparent and if the value is 3 the pixel will be an inversion of the underlying screen color.

This solid fill option is not yet available for this function.

Parameters:

DevID	- a registered device ID
xc,yc	- center point for the ellipse (in pixels)
xr	- horizontal radius of the ellipse (in pixels)
yr	- vertical radius of the ellipse (in pixels)
Color	- a two bit value (0 to 3) to draw the rectangle with
SolidFill	- flag to enable filling the interior of the ellipse (not used)

Return Value: ERR_OK - operation completed with no problems

int seDrawInkCircle(int DevID, long x, long y, long Radius, DWORD Color, BOOL Solid-Fill)

Description: This routine draws a circle in the ink layer display buffer. The center of the circle will be at x,y and the circle will have a radius of 'Radius' pixels.

The value of 'Color' must be 0 to 3. Values 0 and 1 refer to the two user definable colors. If 'Color' is 2 then the pixel will be transparent and if the value is 3 the pixel will be an inversion of the underlying screen color.

Currently seDrawCursorCircle() does not support the solid fill option.

Parameters:

DevID	- a registered device ID
x,y	- center of the circle (in pixels)
Radius	- circle radius (in pixels)
Color	- a two bit (0 to 3) value to draw the circle with
SolidFill	- flag to fill the interior of the circle (not used)

Return Value: ERR_OK - operation completed with no problems

Power Save

This section covers the HAL functions dealing with the Power Save features of the S1D13505.

int seSWSuspend(int DevID, BOOL Suspend)

Description: Causes the S1D13505 to enter software suspend mode.

When software suspend mode is engaged the display is disabled and display buffer is inaccessible. In this mode the registers and the LUT are accessible.

Parameters:

DevID	- a registered device ID
Suspend	- boolean flag to indicate which state to engage. - enter suspend mode when non-zero and return to normal power when equal to zero.

Return Value: ERR_OK - operation completed with no problems

int seHWSuspend(int DevID, BOOL Suspend)

Description: Causes the S1D13505 to enter/leave hardware suspend mode. This option is only supported on S5U13505P00C ISA evaluation boards.

When hardware suspend mode is engaged the display is disabled and display buffer is inaccessible and the registers and LUT are inaccessible.

Parameters:

DevID	- a registered device ID
Suspend	- boolean flag to indicate which state to engage. - enter suspend mode when non-zero and return to normal power when equal to zero.

Return Value: ERR_OK - operation completed with no problems

X-LIB Support**int seGetLinearDispAddr(int device, DWORD * pDispLogicalAddr)**

Description: Determines the logical address of the start of the display buffer. This address may be used in programs for direct control over the display buffer.

Parameter: device - registered device ID
pDispLogicalAddr - logical address is returned in this variable

Return Value: ERR_OK - operation completed with no problems

12 SAMPLE CODE

12.1 Introduction

There are two included examples of programming the S1D13505 color graphics controller. First is a demonstration using the HAL library and the second without. These code samples are for example purposes only. Lastly, are three header files that may make some of the structures used clearer.

Sample Code Using the 13505HAL API

```

/*
// Sample code using 1355HAL API
*/

/*
**-----
**
** Created 1998, Epson Research & Development
** Vancouver Design Centre
** Copyright (c) Seiko Epson Corp. 1998. All rights reserved.
**
** The HAL API code is configured for the following:
**
** 25.175 MHz ClkI
** 640x480 8 bit dual color STN panel @60Hz
** 50 ns EDO, 32 ms (self) refresh time
** Initial color depth - 8 bpp
**-----
*/

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "hal.h"                                     /
* Structures, constants and prototypes. */          /
#include "appcfg.h"
* HAL configuration information. */

/*-----*/
void main(void)
{
    int ChipId;
    int Device;

    /*
    ** Initialize the HAL.
    ** This step sets up the HAL for use but does not access the 1355.
    */
    switch (seRegisterDevice(&HalInfo, &Device))
    {
        case ERR_OK:
            break;
        case HAL_DEVICE_ERR:
            printf("\nERROR: Too many devices registered.");
            exit(1);
        default:
            printf("\nERROR: Could not register SED1355 device.");
            exit(1);
    }

    /*
    ** Identify that this is indeed an SED1355.
    */
    seGetId( Device, &ChipId);
    if (ID_S1D13505F00A != ChipId)
    {
        printf("\nERROR: Did not detect SED1355.");
        exit(1);
    }

    /*
    ** Initialize the SED1355.

```

```
** This step will actually program the registers with values taken from
** the default register table in appcfg.h.
*/
if (ERR_OK != seSetInit(Device))
{
    printf("\nERROR: Could not initialize device.");
    exit(1);
}

/*
** The default initialization clears the display.
** Draw a 100x100 red rectangle in the upper left corner (0,0)
** of the display.
*/
seDrawRect(Device, 0, 0, 100, 100, 1, TRUE);

/*
** Init the HW cursor. The HAL performs several calculations to
** determine the best location to place the cursor image and
** will use that location from here on.
** The background must be set to transparent.
*/
seInitCursor(Device);
seDrawCursorRect(Device, 0, 0, 63, 63, 2, TRUE);

/*
** Set the first user definable color to black and
** the second user definable color to white.
*/
seSetCursorColor(Device, 0, 0);
seSetCursorColor(Device, 1, 0xFFFFFFFF);

/*
** Draw a hollow rectangle around the cursor and move
** the cursor to 101,101.
*/
seDrawCursorRect(Device, 0, 0, 63, 63, 1, FALSE);
seMoveCursor(Device, 101, 101);
exit(0);
```

Sample Code Without Using the 13505HAL API

```

/*
=====
** INIT13505.C - sample code demonstrating the initialization of the SED1355.
** Beta release 2.0 98-10-29
**
** The code in this example will perform initialization to the following
** specification:
**
** - 640 x 480 dual 16-bit color passive panel.
** - 75 Hz frame rate.
** - 8 BPP (256 colors).
** - 33 MHz input clock.
** - 2 MB of 60 ns EDO memory.
**
** *** This is sample code only! ***
**
** This means:
** 1) Generic C is used. I assume that pointers can access the
**    relevant memory addresses (this is not always the case).
**    i.e. using the 1355B0B card on an Intel 16 bit platform will require
**    changes to use a DOS extender to access memory and registers.
** 2) Register setup is done with discrete writes rather than being
**    table driven. This allows for clearer commenting. A real program
**    would probably store the register settings in an array and loop
**    through the array writing each element to a control register.
** 3) The pointer assignment for the register offset does not work on
**    Intel 16 bit platforms.
**
** -----
** Copyright (c) 1998 Epson Research and Development, Inc.
** All Rights Reserved.
** =====
*/
/*
** Note that only the upper four bits of the LUT are actually used.
*/
unsigned char LUT8[256*3] =
{
/* Primary and secondary colors */
0x00, 0x00, 0x00, 0x00, 0x00, 0xA0, 0x00, 0x00, 0xA0, 0x00, 0x00, 0xA0, 0x00, 0xA0, 0xA0,
0xA0, 0x00, 0x00, 0xA0, 0x00, 0xA0, 0xA0, 0xA0, 0xA0, 0x00, 0xA0, 0xA0, 0xA0, 0xA0, 0xA0,
0x50, 0x50, 0x50, 0x00, 0x00, 0xF0, 0x00, 0xF0, 0x00, 0x00, 0xF0, 0xF0,
0xF0, 0x00, 0x00, 0xF0, 0xF0, 0xF0, 0xF0, 0xF0, 0xF0, 0xF0, 0xF0, 0xF0,
/* Gray shades */
0x00, 0x00, 0x00, 0x10, 0x10, 0x10, 0x20, 0x20, 0x20, 0x30, 0x30, 0x30,
0x40, 0x40, 0x40, 0x50, 0x50, 0x50, 0x60, 0x60, 0x60, 0x70, 0x70, 0x70,
0x80, 0x80, 0x80, 0x90, 0x90, 0x90, 0xA0, 0xA0, 0xA0, 0xB0, 0xB0, 0xB0,
0xC0, 0xC0, 0xC0, 0xD0, 0xD0, 0xD0, 0xE0, 0xE0, 0xE0, 0xF0, 0xF0, 0xF0,
/* Black to red */
0x00, 0x00, 0x00, 0x10, 0x00, 0x00, 0x20, 0x00, 0x00, 0x30, 0x00, 0x00,
0x40, 0x00, 0x00, 0x50, 0x00, 0x00, 0x60, 0x00, 0x00, 0x70, 0x00, 0x00,
0x80, 0x00, 0x00, 0x90, 0x00, 0x00, 0xA0, 0x00, 0x00, 0xB0, 0x00, 0x00,
0xC0, 0x00, 0x00, 0xD0, 0x00, 0x00, 0xE0, 0x00, 0x00, 0xF0, 0x00, 0x00,
/* Black to green */
0x00, 0x00, 0x00, 0x00, 0x10, 0x00, 0x00, 0x20, 0x00, 0x00, 0x30, 0x00,
0x00, 0x40, 0x00, 0x00, 0x50, 0x00, 0x00, 0x60, 0x00, 0x00, 0x70, 0x00,
0x00, 0x80, 0x00, 0x00, 0x90, 0x00, 0x00, 0xA0, 0x00, 0x00, 0xB0, 0x00,
0x00, 0xC0, 0x00, 0x00, 0xD0, 0x00, 0x00, 0xE0, 0x00, 0x00, 0xF0, 0x00,
/* Black to blue */
0x00, 0x00, 0x00, 0x00, 0x00, 0x10, 0x00, 0x00, 0x20, 0x00, 0x00, 0x30,
0x00, 0x00, 0x40, 0x00, 0x00, 0x50, 0x00, 0x00, 0x60, 0x00, 0x00, 0x70,
0x00, 0x00, 0x80, 0x00, 0x00, 0x90, 0x00, 0x00, 0xA0, 0x00, 0x00, 0xB0,
0x00, 0x00, 0xC0, 0x00, 0x00, 0xD0, 0x00, 0x00, 0xE0, 0x00, 0x00, 0xF0,
/* Blue to cyan (blue and green) */
0x00, 0x00, 0xF0, 0x00, 0x10, 0xF0, 0x00, 0x20, 0xF0, 0x00, 0x30, 0xF0,
0x00, 0x40, 0xF0, 0x00, 0x50, 0xF0, 0x00, 0x60, 0xF0, 0x00, 0x70, 0xF0,
0x00, 0x80, 0xF0, 0x00, 0x90, 0xF0, 0x00, 0xA0, 0xF0, 0x00, 0xB0, 0xF0,
0x00, 0xC0, 0xF0, 0x00, 0xD0, 0xF0, 0x00, 0xE0, 0xF0, 0x00, 0xF0, 0xF0,
/* Cyan (blue and green) to green */
0x00, 0xF0, 0xF0, 0x00, 0xF0, 0xE0, 0x00, 0xF0, 0xD0, 0x00, 0xF0, 0xC0,
0x00, 0xF0, 0xB0, 0x00, 0xF0, 0xA0, 0x00, 0xF0, 0x90, 0x00, 0xF0, 0x80,
0x00, 0xF0, 0x70, 0x00, 0xF0, 0x60, 0x00, 0xF0, 0x50, 0x00, 0xF0, 0x40,
0x00, 0xF0, 0x30, 0x00, 0xF0, 0x20, 0x00, 0xF0, 0x10, 0x00, 0xF0, 0x00,
/* Green to yellow (red and green) */
0x00, 0xF0, 0x00, 0x10, 0xF0, 0x00, 0x20, 0xF0, 0x00, 0x30, 0xF0, 0x00,
0x40, 0xF0, 0x00, 0x50, 0xF0, 0x00, 0x60, 0xF0, 0x00, 0x70, 0xF0, 0x00,
0x80, 0xF0, 0x00, 0x90, 0xF0, 0x00, 0xA0, 0xF0, 0x00, 0xB0, 0xF0, 0x00,

```

```

0xC0, 0xF0, 0x00, 0xD0, 0xF0, 0x00, 0xE0, 0xF0, 0x00, 0xF0, 0xF0, 0x00,
/* Yellow (red and green) to red */
0xF0, 0xF0, 0x00, 0xF0, 0xE0, 0x00, 0xF0, 0xD0, 0x00, 0xF0, 0xC0, 0x00,
0xF0, 0xB0, 0x00, 0xF0, 0xA0, 0x00, 0xF0, 0x90, 0x00, 0xF0, 0x80, 0x00,
0xF0, 0x70, 0x00, 0xF0, 0x60, 0x00, 0xF0, 0x50, 0x00, 0xF0, 0x40, 0x00,
0xF0, 0x30, 0x00, 0xF0, 0x20, 0x00, 0xF0, 0x10, 0x00, 0xF0, 0x00, 0x00,
/* Red to magenta (blue and red) */
0xF0, 0x00, 0x00, 0xF0, 0x00, 0x10, 0xF0, 0x00, 0x20, 0xF0, 0x00, 0x30,
0xF0, 0x00, 0x40, 0xF0, 0x00, 0x50, 0xF0, 0x00, 0x60, 0xF0, 0x00, 0x70,
0xF0, 0x00, 0x80, 0xF0, 0x00, 0x90, 0xF0, 0x00, 0xA0, 0xF0, 0x00, 0xB0,
0xF0, 0x00, 0xC0, 0xF0, 0x00, 0xD0, 0xF0, 0x00, 0xE0, 0xF0, 0x00, 0xF0,
/* Magenta (blue and red) to blue */
0xF0, 0x00, 0xF0, 0xE0, 0x00, 0xF0, 0xD0, 0x00, 0xF0, 0xC0, 0x00, 0xF0,
0xB0, 0x00, 0xF0, 0xA0, 0x00, 0xF0, 0x90, 0x00, 0xF0, 0x80, 0x00, 0xF0,
0x70, 0x00, 0xF0, 0x60, 0x00, 0xF0, 0x50, 0x00, 0xF0, 0x40, 0x00, 0xF0,
0x30, 0x00, 0xF0, 0x20, 0x00, 0xF0, 0x10, 0x00, 0xF0, 0x00, 0x00, 0xF0,
/* Black to magenta (blue and red) */
0x00, 0x00, 0x00, 0x10, 0x00, 0x10, 0x20, 0x00, 0x20, 0x30, 0x00, 0x30,
0x40, 0x00, 0x40, 0x50, 0x00, 0x50, 0x60, 0x00, 0x60, 0x70, 0x00, 0x70,
0x80, 0x00, 0x80, 0x90, 0x00, 0x90, 0xA0, 0x00, 0xA0, 0xB0, 0x00, 0xB0,
0xC0, 0x00, 0xC0, 0xD0, 0x00, 0xD0, 0xE0, 0x00, 0xE0, 0xF0, 0x00, 0xF0,
/* Black to cyan (blue and green) */
0x00, 0x00, 0x00, 0x00, 0x10, 0x10, 0x00, 0x20, 0x20, 0x00, 0x30, 0x30,
0x00, 0x40, 0x40, 0x00, 0x50, 0x50, 0x00, 0x60, 0x60, 0x00, 0x70, 0x70,
0x00, 0x80, 0x80, 0x00, 0x90, 0x90, 0x00, 0xA0, 0xA0, 0x00, 0xB0, 0xB0,
0x00, 0xC0, 0xC0, 0x00, 0xD0, 0xD0, 0x00, 0xE0, 0xE0, 0x00, 0xF0, 0xF0,
/* Red to white */
0xF0, 0x00, 0x00, 0xF0, 0x10, 0x10, 0xF0, 0x20, 0x20, 0xF0, 0x30, 0x30,
0xF0, 0x40, 0x40, 0xF0, 0x50, 0x50, 0xF0, 0x60, 0x60, 0xF0, 0x70, 0x70,
0xF0, 0x80, 0x80, 0xF0, 0x90, 0x90, 0xF0, 0xA0, 0xA0, 0xF0, 0xB0, 0xB0,
0xF0, 0xC0, 0xC0, 0xF0, 0xD0, 0xD0, 0xF0, 0xE0, 0xE0, 0xF0, 0xF0, 0xF0,
/* Green to white */
0x00, 0xF0, 0x00, 0x10, 0xF0, 0x10, 0x20, 0xF0, 0x20, 0x30, 0xF0, 0x30,
0x40, 0xF0, 0x40, 0x50, 0xF0, 0x50, 0x60, 0xF0, 0x60, 0x70, 0xF0, 0x70,
0x80, 0xF0, 0x80, 0x90, 0xF0, 0x90, 0xA0, 0xF0, 0xA0, 0xB0, 0xF0, 0xB0,
0xC0, 0xF0, 0xC0, 0xD0, 0xF0, 0xD0, 0xE0, 0xF0, 0xE0, 0xF0, 0xF0, 0xF0,
/* Blue to white */
0x00, 0x00, 0xF0, 0x10, 0x10, 0xF0, 0x20, 0x20, 0xF0, 0x30, 0x30, 0xF0,
0x40, 0x40, 0xF0, 0x50, 0x50, 0xF0, 0x60, 0x60, 0xF0, 0x70, 0x70, 0xF0,
0x80, 0x80, 0xF0, 0x90, 0x90, 0xF0, 0xA0, 0xA0, 0xF0, 0xB0, 0xB0, 0xF0,
0xC0, 0xC0, 0xF0, 0xD0, 0xD0, 0xF0, 0xE0, 0xE0, 0xF0, 0xF0, 0xF0, 0xF0
};

/*
** REGISTER_OFFSET points to the starting address of the SED1355 registers
*/
#define REGISTER_OFFSET ((unsigned char *) 0x14000000)
/*
** DISP_MEM_OFFSET points to the starting address of the display buffer memory
*/
#define DISP_MEM_OFFSET ((unsigned char *) 0x4000000)
/*
** DISP_MEMORY_SIZE is the size of display buffer memory
*/
#define DISP_MEMORY_SIZE 0x200000
/*
** Calculate the value to put in Ink/Cursor Start Address Select Register
** Offset = (DISP_MEM_SIZE - (X * 8192))
** We want the offset to be just past the end of display memory so:
** (640 * 480) = DISP_MEMORY_SIZE - (X * 8192)
**
** CURSOR_START = (DISP_MEMORY_SIZE - (640 * 480)) / 8192
*/
#define CURSOR_START 218
void main(void)
{
    unsigned char * pRegs = REGISTER_OFFSET;
    unsigned char * pMem;
    unsigned char * pLUT;
    unsigned char * pTmp;
    unsigned char * pCursor;
    long lpCnt;
    int idx;
    int rgb;
    long x, y;
    /*
    ** Initialize the chip.
    */
    /*
    ** Step 1: Enable the host interface.
    */

```



```

**
** Register 1B: Miscellaneous Disable - host interface enabled, half frame
**             buffer enabled.
**
*/
*(pRegs + 0x1B) = 0x00;           /* 0000 0000 */
/*
** Step 2: Disable the FIFO
**
*/
*(pRegs + 0x23) = 0x80;           /* 1000 0000 */
/*
** Step 3: Set Memory Configuration
**
** Register 1: Memory Configuration - 4 ms refresh, EDO
**
*/
*(pRegs + 0x01) = 0x30;           /* 0011 0000 */
/*
** Step 4: Set Performance Enhancement 0 register
**
*/
*(pRegs + 0x22) = 0x24;           /* 0010 0100 */
/*
** Step 5: Set the rest of the registers in order.
**
*/
/*
** Register 2: Panel Type - 16-bit, format 1, color, dual, passive.
**
*/
*(pRegs + 0x02) = 0x26;           /* 0010 0110 */
/*
** Register 3: Mod Rate
**
*/
*(pRegs + 0x03) = 0x00;           /* 0000 0000 */
/*
** Register 4: Horizontal Display Width (HDP) - 640 pixels
**             (640 / 8) - 1 = 79t = 4Fh
**
*/
*(pRegs + 0x04) = 0x4f;           /* 0100 1111 */
/*
** Register 5: Horizontal Non-Display Period (HNDP)
**             PCLK
**             Frame Rate = -----
**                       (HDP + HNDP) * (VDP + VNDP)
**
**                       16,500,000
**             = -----
**             (640 + HNDP) * (480 + VNDP)
**
** HNDP and VNDP must be calculated such that the desired frame rate
** is achieved.
**
*/
*(pRegs + 0x05) = 0x1f;           /* 0001 1111 */
/*
** Register 6: HRTC/FPLINE Start Position - applicable to CRT/TFT only.
**
*/
*(pRegs + 0x06) = 0x00;           /* 0000 0000 */
/*
** Register 7: HRTC/FPLINE Pulse Width - applicable to CRT/TFT only.
**
*/
*(pRegs + 0x07) = 0x00;           /* 0000 0000 */
/*
** Registers 8-9: Vertical Display Height (VDP) - 480 lines.
**             480/2 - 1 = 239t = 0xEF
**
*/
*(pRegs + 0x08) = 0xEF;           /* 1110 1111 */
*(pRegs + 0x09) = 0x00;           /* 0000 0000 */
/*
** Register A: Vertical Non-Display Period (VNDP)
**             This register must be programed with register 5 (HNDP)
**             to arrive at the frame rate closest to the desired
**             frame rate.
**
*/
*(pRegs + 0x0A) = 0x01;           /* 0000 0001 */
/*
** Register B: VRTC/FPFRAME Start Position - applicable to CRT/TFT only.
**
*/
*(pRegs + 0x0B) = 0x00;           /* 0000 0000 */
/*
** Register C: VRTC/FPFRAME Pulse Width - applicable to CRT/TFT only.
**
*/
*(pRegs + 0x0C) = 0x00;           /* 0000 0000 */
/*
** Register D: Display Mode - 8 BPP, LCD disabled.

```

```

*/
*(pRegs + 0x0D) = 0x0C;           /* 0000 1100 */
/*
** Registers E-F: Screen 1 Line Compare - unless setting up for
** split screen operation use 0x3FF.
*/
*(pRegs + 0x0E) = 0xFF;           /* 1111 1111 */
*(pRegs + 0x0F) = 0x03;           /* 0000 0011 */
/*
** Registers 10-12: Screen 1 Display Start Address - start at the
** first byte in display memory.
*/
*(pRegs + 0x10) = 0x00;           /* 0000 0000 */
*(pRegs + 0x11) = 0x00;           /* 0000 0000 */
*(pRegs + 0x12) = 0x00;           /* 0000 0000 */
/*
** Register 13-15: Screen 2 Display Start Address - not applicable
** unless setting up for split screen operation.
*/
*(pRegs + 0x13) = 0x00;           /* 0000 0000 */
*(pRegs + 0x14) = 0x00;           /* 0000 0000 */
*(pRegs + 0x15) = 0x00;           /* 0000 0000 */
/*
** Register 16-17: Memory Address Offset - this address represents the
** starting WORD. At 8BPP our 640 pixel width is 320
** WORDS
*/
*(pRegs + 0x16) = 0x40;           /* 0100 0000 */
*(pRegs + 0x17) = 0x01;           /* 0000 0001 */
/*
** Register 18: Pixel Panning
*/
*(pRegs + 0x18) = 0x00;           /* 0000 0000 */
/*
** Register 19: Clock Configuration - In this case we must divide
** PCLK by 2 to arrive at the best frequency to set
** our desired panel frame rate.
*/
*(pRegs + 0x19) = 0x01;           /* 0000 0001 */
/*
** Register 1A: Power Save Configuration - enable LCD power, CBR refresh,
** not suspended.
*/
*(pRegs + 0x1A) = 0x00;           /* 0000 0000 */
/*
** Register 1C-1D: MD Configuration Readback - these registers are
** read only, but it's OK to write a 0 to keep
** the register configuration logic simpler.
*/
*(pRegs + 0x1C) = 0x00;           /* 0000 0000 */
*(pRegs + 0x1D) = 0x00;           /* 0000 0000 */
/*
** Register 1E-1F: General I/O Pins Configuration
*/
*(pRegs + 0x1E) = 0x00;           /* 0000 0000 */
*(pRegs + 0x1F) = 0x00;           /* 0000 0000 */
/*
** Register 20-21: General I/O Pins Control
*/
*(pRegs + 0x20) = 0x00;           /* 0000 0000 */
*(pRegs + 0x21) = 0x00;           /* 0000 0000 */
/*
** Registers 24-26: LUT control.
** For this example do a typical 8 BPP LUT setup.
**
** Setup the pointer to the LUT data and reset the LUT index register.
** Then, loop writing each of the RGB LUT data elements.
*/
pLUT = LUT8;
*(pRegs + 0x24) = 0;
for (idx = 0; idx < 256; idx++)
{
    for (rgb = 0; rgb < 3; rgb++)
    {
        *(pRegs + 0x26) = *pLUT;
        pLUT++;
    }
}
/*
** Register 27: Ink/Cursor Control - disable ink/cursor

```

```

*/
*(pRegs + 0x27) = 0x00;          /* 0000 0000 */
/*
** Registers 28-29: Cursor X Position
*/
*(pRegs + 0x28) = 0x00;          /* 0000 0000 */
*(pRegs + 0x29) = 0x00;          /* 0000 0000 */
/*
** Registers 2A-2B: Cursor Y Position
*/
*(pRegs + 0x2A) = 0x00;          /* 0000 0000 */
*(pRegs + 0x2B) = 0x00;          /* 0000 0000 */
/*
** Registers 2C-2D: Ink/Cursor Color 0 - blue
*/
*(pRegs + 0x2C) = 0x1F;          /* 0001 1111 */
*(pRegs + 0x2D) = 0x00;          /* 0000 0000 */
/*
** Registers 2E-2F: Ink/Cursor Color 1 - green
*/
*(pRegs + 0x2E) = 0xE0;          /* 1110 0000 */
*(pRegs + 0x2F) = 0x07;          /* 0000 0111 */
/*
** Register 30: Ink/Cursor Start Address Select
*/
*(pRegs + 0x30) = 0x00;          /* 0000 0000 */
/*
** Register 31: Alternate FRM Register
*/
*(pRegs + 0x31) = 0x00;
/*
** Register 23: Performance Enhancement - display FIFO enabled, optimum
**               performance. The FIFO threshold is set to 0x00; for
**               15/16 bpp modes, set the FIFO threshold
**               to a higher value, such as 0x1B.
*/
*(pRegs + 0x23) = 0x00;          /* 0000 0000 */
/*
** Register D: Display Mode - 8 BPP, LCD enable.
*/
*(pRegs + 0x0D) = 0x0D;          /* 0000 1101 */
/*
** Clear memory by filling 2 MB with 0
*/
pMem = DISP_MEM_OFFSET;
for (lpCnt = 0; lpCnt < DISP_MEMORY_SIZE; lpCnt++)
{
    *pMem = 0;
    pMem++;
}
/*
** Draw a 100x100 red rectangle in the upper left corner (0, 0)
** of the display.
*/
pMem = DISP_MEM_OFFSET;
for (y = 0; y < 100; y++)
{
    pTmp = pMem + y * 640L;
    for (x = 0; x < 100; x++)
    {
        *pTmp = 0x0c;
        pTmp++;
    }
}
/*
** Init the HW cursor. In this example the cursor memory will be located
** immediately after display memory. Why here? Because it's an easy
** location to calculate and will not interfere with the half frame buffer.
** Additionally, the HW cursor can be turned into an ink layer quite
** easily from this location.
*/
*(pRegs + 0x30) = CURSOR_START;
pTmp = pCursor = pMem + (DISP_MEMORY_SIZE - (CURSOR_START * 8192L));
/*
** Set the contents of the cursor memory such that the cursor
** is transparent. To do so, write a 10101010b pattern in each byte.
** The cursor is 2 bpp so a 64x64 cursor requires
** 64/4 * 64 = 1024 bytes of memory.
*/
for (lpCnt = 0; lpCnt < 1024; lpCnt++)

```

```
{
    *pTmp = 0xAA;
    pTmp++;
}
/*
** Set the first user definable cursor color to black and
** the second user definable cursor color to white.
*/
*(pRegs + 0x2C) = 0;
*(pRegs + 0x2D) = 0;
*(pRegs + 0x2E) = 0xFF;
*(pRegs + 0x2F) = 0xFF;
/*
** Draw a hollow rectangle around the cursor.
*/
pTmp = pCursor;
for (lpCnt = 0; lpCnt < 16; lpCnt++)
{
    *pTmp = 0x55;
    pTmp++;
}
for (lpCnt = 0; lpCnt < 14; lpCnt++)
{
    *pTmp = 0x6A;
    pTmp += 15;
    *pTmp = 0xA9;
    pTmp++;
}
for (lpCnt = 0; lpCnt < 16; lpCnt++)
{
    *pTmp = 0x55;
    pTmp++;
}
/*
** Move the cursor to 100, 100.
*/
/*
** First we wait for the next vertical non-display
** period before updating the position registers.
*/
while (*(pRegs + 0x0A) & 0x80); /* wait while in VNDP */
while (!(*(pRegs + 0x0A) & 0x80)); /* wait while in VDP */
/*
** Now update the position registers.
*/
*(pRegs + 0x28) = 100; /* Set Cursor X = 100 */
*(pRegs + 0x29) = 0x00;
*(pRegs + 0x2A) = 100; /* Set Cursor Y = 100 */
*(pRegs + 0x2B) = 0x00;
/*
** Enable the hardware cursor.
*/
*(pRegs + 0x27) = 0x40;
}
```

Header Files

The following header files are included as they help to explain some of the structures used when programming the S1D13505.

The following header file defines the structure used to store the configuration information contained in all utilities using the S1D13505HAL API.

```

/*****
/* 1355 HAL INF          (do not remove)          */
/* HAL_STRUCT Information generated by 1355CFG.EXE */
/* Copyright (c) 1998 Seiko Epson Corp. All rights reserved. */
/* Include this file ONCE in your primary source file */
*****/

HAL_STRUCT HalInfo =
{
  "1355 HAL EXE",      /* ID string */
  0x1234,              /* Detect Endian */
  sizeof(HAL_STRUCT), /* Size */
  0,                  /* Default Mode */

  {
    { /* LCD */
      0x00, 0x50, 0x16, 0x00, 0x4F, 0x03, 0x00, 0x00,
      0xEF, 0x00, 0x34, 0x00, 0x00, 0x0D, 0xFF, 0x03,
      0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x40, 0x01,
      0x00, 0x01, 0x02, 0x00, 0x00, 0x00, 0x00, 0x00,
      0x00, 0x00, 0x48, 0x00, 0x00, 0x00, 0x00, 0x00,
      0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
      0x00, 0x00
    },

    { /* CRT */
      0x00, 0x50, 0x16, 0x00, 0x4F, 0x13, 0x01, 0x0B,
      0xDF, 0x01, 0x2B, 0x09, 0x01, 0x0E, 0xFF, 0x03,
      0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x40, 0x01,
      0x00, 0x00, 0x02, 0x01, 0x00, 0x00, 0x00, 0x00,
      0x00, 0x00, 0x48, 0x00, 0x00, 0x00, 0x00, 0x00,
      0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
      0x00, 0x00
    },

    { /* SIMUL */
      0xFF, 0x50, 0x16, 0x00, 0x4F, 0x13, 0x01, 0x0B,
      0xDF, 0x01, 0x2B, 0x09, 0x01, 0x0F, 0xFF, 0x03,
      0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x40, 0x01,
      0x00, 0x01, 0x02, 0x01, 0x00, 0x00, 0x00, 0x00,
      0x00, 0x00, 0x48, 0x00, 0x00, 0x00, 0x00, 0x00,
      0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
      0x00, 0x00
    },
  },

  25175, /* ClkI (kHz) */
  8000, /* BusClk (kHz) */
  0xE00000, /* Register Address */
  0xC00000, /* Display Address */
  60, /* Panel Frame Rate (Hz) */
  60, /* CRT Frame Rate (Hz) */
  50, /* Memory speed in ns */
  84, /* Ras to Cas Delay in ns */
  30, /* Ras Access Charge time in ns */
  50, /* RAS Access Charge time in ns */
  16, /* Host CPU bus width in bits */
};
/*

```

The following header file defines the S1D13505HAL registers.

```

/*=====
**  HAL_REGS.H
**  -----
**  Created 1998, Epson Research & Development
**      Vancouver Design Center.
**  Copyright(c) Seiko Epson Corp. 1997, 1998. All rights reserved.
**  -----
**
**  $Header:      $
**
**  $Revision:    $
**
**  $Log:         $
**
**===== */

#ifndef __HAL_REGS_H__
#define __HAL_REGS_H__

/*
** 1355 register names
*/

#define REG_REVISION_CODE          0x00
#define REG_MEMORY_CONFIG         0x01
#define REG_PANEL_TYPE            0x02
#define REG_MOD_RATE              0x03
#define REG_HORZ_DISP_WIDTH       0x04
#define REG_HORZ_NONDISP_PERIOD   0x05
#define REG_HRTC_START_POSITION   0x06
#define REG_HRTC_PULSE_WIDTH      0x07
#define REG_VERT_DISP_HEIGHT0     0x08
#define REG_VERT_DISP_HEIGHT1     0x09
#define REG_VERT_NONDISP_PERIOD   0x0A
#define REG_VRTC_START_POSITION   0x0B
#define REG_VRTC_PULSE_WIDTH      0x0C
#define REG_DISPLAY_MODE          0x0D
#define REG_SCRN1_LINE_COMPARE0   0x0E
#define REG_SCRN1_LINE_COMPARE1   0x0F
#define REG_SCRN1_DISP_START_ADDR0 0x10
#define REG_SCRN1_DISP_START_ADDR1 0x11
#define REG_SCRN1_DISP_START_ADDR2 0x12
#define REG_SCRN2_DISP_START_ADDR0 0x13
#define REG_SCRN2_DISP_START_ADDR1 0x14
#define REG_SCRN2_DISP_START_ADDR2 0x15
#define REG_MEM_ADDR_OFFSET0      0x16
#define REG_MEM_ADDR_OFFSET1      0x17
#define REG_PIXEL_PANNING         0x18
#define REG_CLOCK_CONFIG          0x19
#define REG_POWER_SAVE_CONFIG     0x1A
#define REG_MISC                  0x1B
#define REG_MD_CONFIG_READBACK0   0x1C
#define REG_MD_CONFIG_READBACK1   0x1D
#define REG_GPIO_CONFIG0          0x1E
#define REG_GPIO_CONFIG1          0x1F
#define REG_GPIO_CONTROL0         0x20
#define REG_GPIO_CONTROL1         0x21
#define REG_PERF_ENHANCEMENT0     0x22
#define REG_PERF_ENHANCEMENT1     0x23
#define REG_LUT_ADDR              0x24
#define REG_RESERVED_1            0x25
#define REG_LUT_DATA              0x26
#define REG_INK_CURSOR_CONTROL    0x27
#define REG_CURSOR_X_POSITION0    0x28
#define REG_CURSOR_X_POSITION1    0x29
#define REG_CURSOR_Y_POSITION0    0x2A
#define REG_CURSOR_Y_POSITION1    0x2B
#define REG_INK_CURSOR_COLOR0_0   0x2C
#define REG_INK_CURSOR_COLOR0_1   0x2D
#define REG_INK_CURSOR_COLOR1_0   0x2E
#define REG_INK_CURSOR_COLOR1_1   0x2F
#define REG_INK_CURSOR_START_ADDR 0x30
#define REG_ALTERNATE_FRM         0x31

/*
** WARNING!!! MAX_REG must be the last available register!!!

```

```

*/
#define MAX_REG                0x31

#endif      /*  __HAL_REGS_H__  */

```

The following header file defines the structures used in the S1D13505HAL API.

```

**=====
** HAL.H
**-----
** Created 1998, Epson Research & Development
**           Vancouver Design Center.
** Copyright(c) Seiko Epson Corp. 1997, 1998. All rights reserved.
**=====
*/

#ifndef __HAL_H_
#define __HAL_H_

#pragma warning(disable:4001)    // Disable the 'single line comment' warning.

#include "hal_regs.h"

/*-----*/

typedef unsigned char  BYTE;
typedef unsigned short WORD;
typedef unsigned long  DWORD;
typedef unsigned int   UINT;
typedef               int   BOOL;

#ifdef INTEL
typedef BYTE  far *LPBYTE;
typedef WORD  far *LPWORD;
typedef DWORD far *LPDWORD;
#else
typedef BYTE      *LPBYTE;
typedef WORD      *LPWORD;
typedef DWORD     *LPDWORD;
#endif

#ifndef LOBYTE
#define LOBYTE(w)    ((BYTE)(w))
#endif

#ifndef HIBYTE
#define HIBYTE(w)    ((BYTE)((((UINT)(w) >> 8) & 0xFF))
#endif

#ifndef LOWORD
#define LOWORD(l)    ((WORD)(DWORD)(l))
#endif

#ifndef HIWORD
#define HIWORD(l)    ((WORD)((((DWORD)(l) >> 16) & 0xFFFF))
#endif

#ifndef MAKEWORD
#define MAKEWORD(lo, hi) ((WORD)((((WORD)(lo) | (((WORD)(hi)) << 8)) )
#endif

#ifndef MAKELONG
#define MAKELONG(lo, hi) ((long)((((WORD)(lo) | (((DWORD)((WORD)(hi))) << 16)))
#endif

#ifndef TRUE
#define TRUE  1
#endif

#ifndef FALSE
#define FALSE 0
#endif

#define OFF 0
#define ON  1

```

```
#ifndef NULL
#ifdef __cplusplus
#define NULL 0
#else
#define NULL ((void *)0)
#endif
#endif

/*-----*/

/*
** SIZE_VERSION is the size of the version string (eg. "1.00")
** SIZE_STATUS is the size of the status string (eg. "b" for beta)
** SIZE_REVISION is the size of the status revision string (eg. "00")
*/
#define SIZE_VERSION 5
#define SIZE_STATUS 2
#define SIZE_REVISION 3

#ifdef ENABLE_DPF /* Debug_printf() */

#define DPF(exp) printf(#exp "\n")
#define DPF1(exp) printf(#exp " = %d\n", exp)
#define DPF2(exp1, exp2) printf(#exp1 "=%d " #exp2 "=%d\n", exp1, exp2)
#define DPFL(exp) printf(#exp " = %x\n", exp)

#else

#define DPF(exp) ((void)0)
#define DPF1(exp) ((void)0)
#define DPFL(exp) ((void)0)

#endif

/*-----*/

enum
{
    ERR_OK = 0, /* No error, call was successful. */
    ERR_FAILED, /* General purpose failure. */

    ERR_UNKNOWN_DEVICE, /* */
    ERR_INVALID_PARAMETER, /* Function was called with invalid parameter. */
    ERR_HAL_BAD_ARG,
    ERR_TOOMANY_DEVS,

    ERR_INVALID_STD_DEVICE
};

/*****
 * Definitions for seGetId()
 *****/
enum
{
    ID_UNKNOWN,
    ID_SED1355,
    ID_SED1355F0A
};

#define MAX_DEVICE 10

/*
** SE_RESERVED is for reserved device
*/
#define SE_RESERVED 0

/*
** DetectEndian is used to determine whether the most significant
** and least significant bytes are reversed by the given compiler.
*/
#define ENDIAN 0x1234
#define REV_ENDIAN 0x3412

/*****
 * Definitions for Internal calculations.
 *****/
```



```

#define MIN_NON_DISP_X      32
#define MAX_NON_DISP_X      256

#define MIN_NON_DISP_Y      2
#define MAX_NON_DISP_Y      64

/*****
 * Definitions for seSetFont
 *****/

enum
{
    HAL_STDOUT,
    HAL_STDIN,
    HAL_DEVICE_ERR
};

#define FONT_NORMAL          0x00
#define FONT_DOUBLE_WIDTH    0x01
#define FONT_DOUBLE_HEIGHT    0x02

enum
{
    RED,
    GREEN,
    BLUE
};

/*****
 * Definitions for seSplitScreen()
 *****/

enum
{
    SCREEN1 = 1,
    SCREEN2
};

/*****
 * Definitions for sePowerSaveMode()
 *****/

#define PWR_CBR_REFRESH      0x00
#define PWR_SELF_REFRESH     0x01
#define PWR_NO_REFRESH       0x02

/*****

enum
{
    DISP_MODE_LCD = 0,
    DISP_MODE_CRT,
    DISP_MODE_SIMULTANEOUS,

    MAX_DISP_MODE
};

typedef struct tagHalStruct
{
    char    szIdString[16];
    WORD    wDetectEndian;
    WORD    wSize;
    WORD    wDefaultMode;

    BYTE    Regs[MAX_DISP_MODE][MAX_REG + 1];

    DWORD    dwClkI;           /* Input Clock Frequency (in kHz) */
    DWORD    dwBusClk;         /* Bus Clock Frequency (in kHz) */
    DWORD    dwRegAddr;        /* Starting address of registers */
    DWORD    dwDispMem;        /* Starting address of display buffer memory */
    WORD     wPanelFrameRate;  /* Desired panel frame rate */

    WORD     wCrtFrameRate;    /* Desired CRT rate */
    WORD     wMemSpeed;        /* Memory speed in ns */
    WORD     wTrc;             /* Ras to Cas Delay in ns */

```

```

WORD   wTrp;                /* Ras Precharge time in ns */
WORD   wTrac;               /* Ras Access Charge time in ns */
WORD   wHostBusWidth;       /* Host CPU bus width in bits */

} HAL_STRUCT;

typedef HAL_STRUCT * PHAL_STRUCT;

#ifdef INTEL
typedef HAL_STRUCT far * LPHAL_STRUCT;
#else
typedef HAL_STRUCT * LPHAL_STRUCT;
#endif

/*=====
/*                               FUNCTION  PROTO-TYPES                               */
/*=====

/*----- HAL Support -----*/

int seInitHal( void );
int seGetDetectedBusWidth(int *bits);
int seRegisterDevice( const LPHAL_STRUCT lpHalInfo, int *Device );
int seGetMemSize( int seReserved1, DWORD *val );

#define CLEAR_MEM            TRUE
#define DONT_CLEAR_MEM      FALSE
int seSetDisplayMode(int device, int DisplayMode, int ClearMem);

int seSetInit(int device);

int seGetId( int seReserved1, int *pId );
void seGetHalVersion( const char **pVersion, const char **pStatus, const char **pStatusRe-
vision );

/*----- Chip Access -----*/

int seGetReg( int seReserved1, int index, BYTE *pValue );
int seSetReg( int seReserved1, int index, BYTE value );

/*----- Misc -----*/

int seSetBitsPerPixel( int seReserved1, UINT nBitsPerPixel );
int seGetBitsPerPixel( int seReserved1, UINT *pBitsPerPixel );

int seGetBytesPerScanline( int seReserved1, UINT *pBytes );
int seGetScreenSize( int seReserved1, UINT *width, UINT *height );
int seHWSuspend(int seReserved1, BOOL val);
int seSelectBusWidth(int seReserved1, int width);

int seDelay( int seReserved1, DWORD Seconds );

int seGetLastUsableByte( int seReserved1, DWORD *LastByte );
int seDisplayEnable(int seReserved1, BYTE NewState);

int seSplitInit( int seReserved1, DWORD wScrn1Addr, DWORD wScrn2Addr );
int seSplitScreen( int nReserved1, int WhichScreen, long VisibleScanlines );
int seVirtInit( int seReserved1, DWORD xVirt, DWORD *yVirt );
int seVirtMove( int seReserved1, int nWhichScreen, DWORD x, DWORD y );

/*----- Power Save -----*/

int seSetPowerSaveMode( int seReserved1, int PowerSaveMode );

/*----- Memory Access -----*/

int seReadDisplayByte( int seReserved1, DWORD offset, BYTE *pByte );
int seReadDisplayWord( int seReserved1, DWORD offset, WORD *pWord );
int seReadDisplayDword( int seReserved1, DWORD offset, DWORD *pDword );

int seWriteDisplayBytes( int seReserved1, DWORD addr, BYTE val, DWORD count );
int seWriteDisplayWords( int seReserved1, DWORD addr, WORD val, DWORD count );
int seWriteDisplayDwords( int seReserved1, DWORD addr, DWORD val, DWORD count );

/*----- Drawing -----*/

int seGetInkStartAddr(int seReserved1, DWORD *addr);

int seGetPixel( int seReserved1, long x, long y, DWORD *pVal );

```

```

int seSetPixel( int seReserved1, long x, long y, DWORD color );
int seDrawLine( int seReserved1, long x1, long y1, long x2, long y2, DWORD color );
int seDrawRect( int seReserved1, long x1, long y1, long x2, long y2, DWORD color, BOOL
SolidFill );
int seDrawEllipse( int seReserved1, long xc, long yc, long xr, long yr, DWORD color, BOOL
SolidFill );
int seDrawCircle( int seReserved1, long xCenter, long yCenter, long radius, DWORD color,
BOOL SolidFill );

/*----- Hardware Cursor -----*/

int seInitCursor(int seReserved1);
int seCursorOff(int seReserved1);
int seGetCursorStartAddr(int seReserved1, DWORD *addr);
int seMoveCursor(int seReserved1, long x, long y);
int seSetCursorColor(int seReserved1, int index, DWORD color);
int seSetCursorPixel( int seReserved1, long x, long y, DWORD color );
int seDrawCursorLine( int seReserved1, long x1, long y1, long x2, long y2, DWORD color );
int seDrawCursorRect( int seReserved1, long x1, long y1, long x2, long y2, DWORD color,
BOOL SolidFill );
int seDrawCursorEllipse( int seReserved1, long xc, long yc, long xr, long yr, DWORD color,
BOOL SolidFill );
int seDrawCursorCircle( int seReserved1, long xCenter, long yCenter, long radius, DWORD
color, BOOL SolidFill );

/*----- Hardware Ink Layer -----*/

int seInitInk(int seReserved1);
int seInkOff(int seReserved1);
int seGetInkStartAddr(int seReserved1, DWORD *addr);
int seSetInkColor(int seReserved1, int index, DWORD color);
int seSetInkPixel( int seReserved1, long x, long y, DWORD color );
int seDrawInkLine( int seReserved1, long x1, long y1, long x2, long y2, DWORD color );
int seDrawInkRect( int seReserved1, long x1, long y1, long x2, long y2, DWORD color, BOOL
SolidFill );
int seDrawInkEllipse( int seReserved1, long xc, long yc, long xr, long yr, DWORD color,
BOOL SolidFill );
int seDrawInkCircle( int seReserved1, long xCenter, long yCenter, long radius, DWORD
color, BOOL SolidFill );

/*----- Color -----*/

int seSetLut( int seReserved1, BYTE *pLut, int count );
int seGetLut( int seReserved1, BYTE *pLut, int count );
int seSetLutEntry( int seReserved1, int index, BYTE *pEntry );
int seGetLutEntry( int seReserved1, int index, BYTE *pEntry );

/*----- C Like Support -----*/

int seDrawText( int seReserved1, char *fmt, ... );
int sePutChar( int seReserved1, int ch );
int seGetChar( void );

/*----- XLIB Support -----*/

int seGetLinearDispAddr(int seReserved1, DWORD *pDispLogicalAddr);
int InitLinear(int seReserved1);

#endif /* _HAL_H_ */

```

APPENDIX SUPPORTED PANEL VALUES

The following tables show related register data for different panels. All the examples are based on 8 bpp and 2M bytes of 50 ns EDO-DRAM.

Table A-1 Passive Single Panel with 40MHz Pixel Clock

Register	Mono 4-Bit 320X240@60Hz	Mono 4-Bit EL 320X240@60Hz	Color 8-Bit 320X240@60Hz	Color 8-Bit Format 2 320X240@60Hz	Notes
REG[02h]	0000 0000	1000 0000	0001 0100	0001 1100	set panel type
REG[03h]	0000 0000	0000 0000	0000 0000	0000 0000	set MOD rate
REG[04h]	0010 0111	0010 0111	0010 0111	0010 0111	set horizontal display width
REG[05h]	0001 0111	0001 0111	0001 0111	0001 0111	set horizontal non-display period
REG[08h]	1110 1111	1110 1111	1110 1111	1110 1111	set vertical display height bits 7-0
REG[09h]	0000 0000	0000 0000	0000 0000	0000 0000	set vertical display height bits 9-8
REG[0Ah]	0011 1110	0011 1110	0011 1110	0011 1110	set vertical non-display period
REG[0Dh]	0000 1101	0000 1101	0000 1101	0000 1101	set 8 bpp and LCD enable
REG[19h]	0000 0011	0000 0011	0000 0011	0000 0011	set MCLK and PCLK divide
REG[1Bh]	0000 0001	0000 0001	0000 0001	0000 0001	disable half frame buffer
REG[24h]	0000 0000	0000 0000	0000 0000	0000 0000	set Look-Up Table address to 0
REG[26h]	load LUT	load LUT	load LUT	load LUT	load Look-Up Table

Register	Mono 8-Bit 640X480@60Hz	Color 8-Bit 640X480@60Hz	Color 16-Bit 640X480@60Hz	Notes
REG[02h]	0001 0000	0001 0100	0010 0100	set panel type
REG[03h]	0000 0000	0000 0000	0000 0000	set MOD rate
REG[04h]	0100 1111	0100 1111	0100 1111	set horizontal display width
REG[05h]	0000 0011	0000 0011	0000 0011	set horizontal non-display period
REG[08h]	1101 1111	1101 1111	1101 1111	set vertical display height bits 7-0
REG[09h]	0000 0001	0000 0001	0000 0001	set vertical display height bits 9-8
REG[0Ah]	0000 0010	0000 0010	0000 0010	set vertical non-display period
REG[0Dh]	0000 1101	0000 1101	0000 1101	set 8 bpp and LCD enable
REG[19h]	0000 0001	0000 0001	0000 0001	set MCLK and PCLK divide
REG[1Bh]	0000 0001	0000 0001	0000 0001	disable half frame buffer
REG[24h]	0000 0000	0000 0000	0000 0000	set Look-Up Table address to 0
REG[26h]	load LUT	load LUT	load LUT	load Look-Up Table

Note: The following settings may not reflect the ideal settings for your system configuration. Power, speed and cost requirements may dictate different starting parameters for your system (e.g. 320 × 240 @ 78 Hz using 12 MHz clock).

Table A-2 Passive Dual Panel with 40MHz Pixel Clock

Register	Mono 4-Bit EL 640X480@60Hz	Mono 8-Bit 640X480@60Hz	Color 8-Bit 640X480@60Hz	Color 16-Bit 640X480@60Hz	Notes
REG[02h]	1000 0010	0001 0010	0001 0110	0010 0110	set panel type
REG[03h]	0000 0000	0000 0000	0000 0000	0000 0000	set MOD rate
REG[04h]	0100 1111	0100 1111	0100 1111	0100 1111	set horizontal display width
REG[05h]	0000 0101	0000 0101	0000 0101	0000 0101	set horizontal non-display period
REG[08h]	1110 1111	1110 1111	1110 1111	1110 1111	set vertical display height bits 7-0
REG[09h]	0000 0000	0000 0000	0000 0000	0000 0000	set vertical display height bits 9-8
REG[0Ah]	0011 1110	0011 1110	0011 1110	0011 1110	set vertical non-display period
REG[0Dh]	0000 1101	0000 1101	0000 1101	0000 1101	set 8 bpp and LCD enable
REG[19h]	0000 0010	0000 0010	0000 0010	0000 0010	set MCLK and PCLK divide
REG[1Bh]	0000 0000	0000 0000	0000 0000	0000 0000	enable half frame buffer
REG[24h]	0000 0000	0000 0000	0000 0000	0000 0000	set Look-Up Table address to 0
REG[26h]	load LUT	load LUT	load LUT	load LUT	load Look-Up Table

Table A-3 TFT Single Panel with 25.175MHz Pixel Clock

Register	Color 16-Bit 640X480@60Hz	Notes
REG[02h]	0010 0101	set panel type
REG[03h]	0000 0000	set MOD rate
REG[04h]	0100 1111	set horizontal display width
REG[05h]	0001 0011	set horizontal non-display period
REG[06h]	0000 0001	set HSYNC start position
REG[07h]	0000 1011	set HSYNC polarity and pulse width
REG[08h]	1101 1111	set vertical display height bits 7-0
REG[09h]	0000 0001	set vertical display height bits 9-8
REG[0Ah]	0010 1011	set vertical non-display period
REG[0Bh]	0000 1001	set VSYNC start position
REG[0Ch]	0000 0001	set VSYNC polarity and pulse width
REG[0Dh]	0000 1101	set 8 bpp and LCD enable
REG[19h]	0000 0000	set MCLK and PCLK divide
REG[1Bh]	0000 0001	disable half frame buffer
REG[24h]	0000 0000	set Look-Up Table address to 0
REG[26h]	load LUT	load Look-Up Table

THIS PAGE IS BLANK.

S1D13505F00A
Embedded RAMDAC LCD/CRT Controller

Utilities

Table of Contents

1	13505CFG CONFIGURATION PROGRAM	3-1
1.1	13505CFG	3-1
	S1D13505 Supported Evaluation Platforms	3-1
	Installation	3-1
	Usage	3-1
1.2	13505CFG Configuration Pages	3-2
	General Page	3-3
	Memory Page	3-4
	Panel Page	3-5
	CRT Page	3-6
	Default Page	3-7
	Open Dialog Box	3-8
	Save As Dialog Box	3-9
	Example	3-10
	Comments	3-11
	Sample Program Messages	3-12
2	13505SHOW DEMONSTRATION PROGRAM	3-13
2.1	13505SHOW	3-13
	S1D13505 Supported Evaluation Platforms	3-13
	Installation	3-13
	Usage	3-14
	13505SHOW Examples	3-14
	Comments	3-15
	Program Messages	3-16
3	13505SPLT DISPLAY UTILITY	3-17
3.1	13505SPLT	3-17
	S1D13505 Supported Evaluation Platforms	3-17
	Installation	3-17
	Usage	3-18
	13505SPLT Example	3-18
	Comments	3-18
	Program Messages	3-19
4	13505VIRT DISPLAY UTILITY	3-20
4.1	13505VIRT	3-20
	S1D13505 Supported Evaluation Platforms	3-20
	Installation	3-20
	Usage	3-21
	13505VIRT Example	3-21
	Comments	3-21
	Program Messages	3-22
5	13505PLAY DIAGNOSTIC UTILITY	3-23
5.1	13505PLAY	3-23
	S1D13505 Supported Evaluation Platforms	3-23
	Installation	3-23
	Usage	3-24
	13505PLAY Example	3-25
	Scripting	3-26
	Comments	3-26
	Program Messages	3-27
6	13505BMP DEMONSTRATION PROGRAM	3-28
6.1	13505BMP	3-28
	S1D13505 Supported Evaluation Platforms	3-28

Installation.....	3-28
Usage	3-28
13505BMP Examples	3-28
Comments	3-29
Program Messages.....	3-29
7 13505PWR SOFTWARE SUSPEND POWER SEQUENCING UTILITY.....	3-30
7.1 13505PWR.....	3-30
S1D13505 Supported Evaluation Platforms	3-30
Installation.....	3-30
Usage	3-31
13505PWR Examples.....	3-31
Comments	3-31
Program Messages.....	3-32

List of Figures

Figure 1-1	General Page	3-3
Figure 1-2	Memory Page	3-4
Figure 1-3	Panel Page	3-5
Figure 1-4	CRT Page.....	3-6
Figure 1-5	Default Page.....	3-7

1 13505CFG CONFIGURATION PROGRAM

1.1 13505CFG

13505CFG is an interactive Windows® '9x program that calculates the S1D13505 register values for a user-defined LCD panel/CRT configuration. The S1D13505 utilities can have their configurations opened, changed, and saved, all from within 13505CFG.

13505CFG is designed to work with the S1D13505 utilities, or any program designed by a software/hardware developer using the Hardware Abstraction Layer (HAL) library. The configuration information can be saved directly into the utility or into a text header file for use by the software/hardware developer.

Note: Seiko Epson does not assume liability for any damage done to the display device as a result of software configuration errors.

S1D13505 Supported Evaluation Platforms

13505CFG only runs on a PC system running Windows '9x.

13505CFG can edit the executable files for the following S1D13505 evaluation platforms:

- PC system with an Intel 80x86 processor.
- M68332BCC (Business Card Computer) board, revision B, with a Motorola MC68332 processor.
- M68EC000IDP (Integrated Development Platform) board, revision 3.0, with a Motorola M68EC000 processor.
- SH3-LCEVB board, revision B, with an Hitachi SH-3 HD6417780 processor.

Installation

Copy the file 13505CFG.EXE to a directory on your hard drive that is in the DOS path.

Usage

In Windows 95, double-click the following icon:



1355Cfg.exe

1.2 13505CFG Configuration Pages

13505CFG provides a series of pages which can be selected by a tab at the top of the main window. The pages are “General”, “Memory”, “Panel”, “CRT”, and “Default”. At the bottom of the window are three buttons: Open, Save As, and Exit.

The basic procedure for using 13505CFG is as follows:

OPEN the configuration values from a current utility (this step is optional).

Change the configuration values as required (see each page description for configuration details).

SAVE the configuration values into the desired utilities, or into an ASCII header file. Each utility must be configured separately.

Note: 13505CFG is designed to work with utilities programmed using a given version of the HAL. If the configuration structure is of a different version, an error message is displayed.

General Page

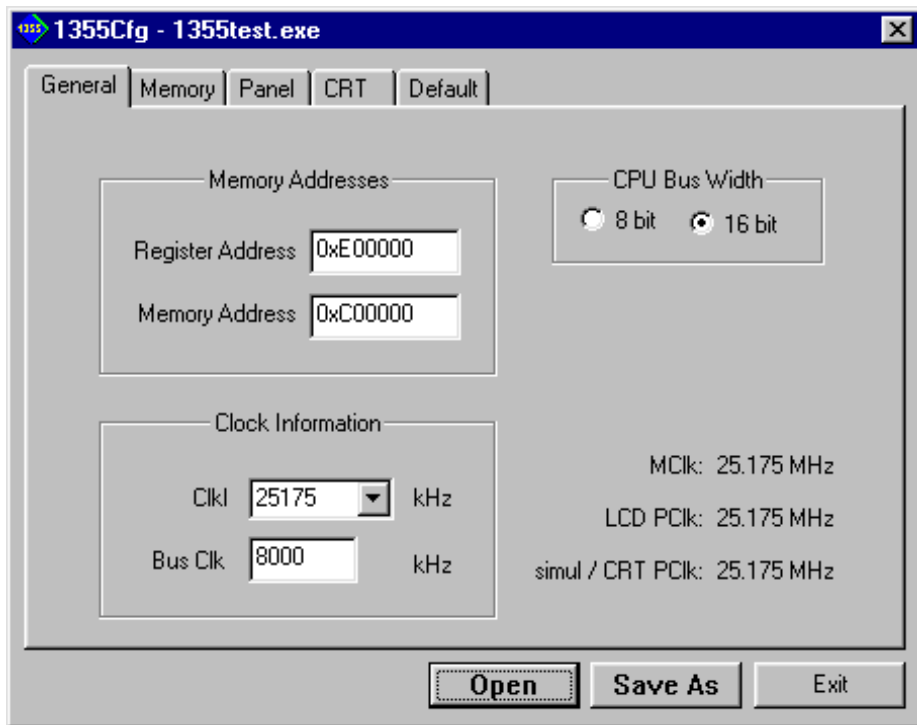


Figure 1-1 General Page

The General Page allows the user to select the following general platform settings:

General Page	
Register Address	Starting address of the registers (in hexadecimal).
Memory Address	Starting address of the display buffer (in hexadecimal).
CPU Bus Width	Host CPU bus width (applicable only to PC).
ClkI	Clock frequency.
Bus Clk	Host bus clock frequency.

Also displayed is the memory clock frequency and the pixel clock frequencies for the following modes: LCD, CRT, and simultaneous display. These clock values will change based on settings on both the General Page and other configuration pages.

These clock frequencies are useful in determining why a particular display mode cannot be set. See the “*SID13505 Hardware Functional Specification*” for more details.

Memory Page

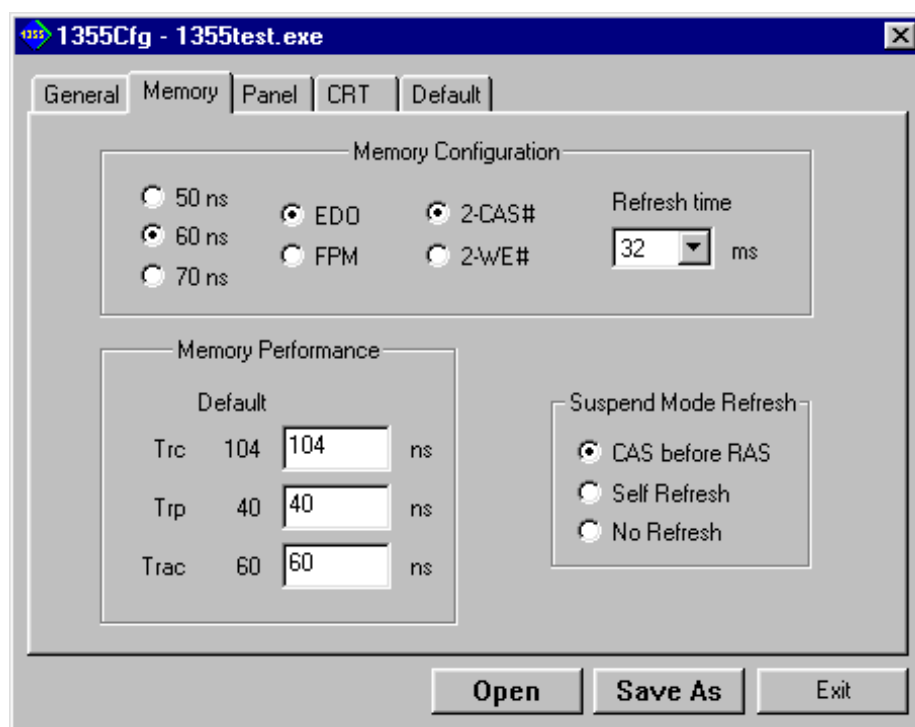


Figure 1-2 Memory Page

The Memory Page allows the user to select the following settings:

Memory Page	
Timing (ns)	Access time for memory.
Memory Type	EDO or FPM.
WE# Control	2CAS# or 2WE#.
Refresh time (ms)	DRAM Refresh Rate (time for 256 refresh cycles).
Trc	Use the values in the DRAM specification. For the S5U13505P00C Evaluation Board, use the values shown in the "Default" column. The values in the "Default" column will change based on the Memory Timing.
Trp	
Trac	
Suspend Mode Refresh	Type of DRAM refresh used in suspend mode.

Panel Page

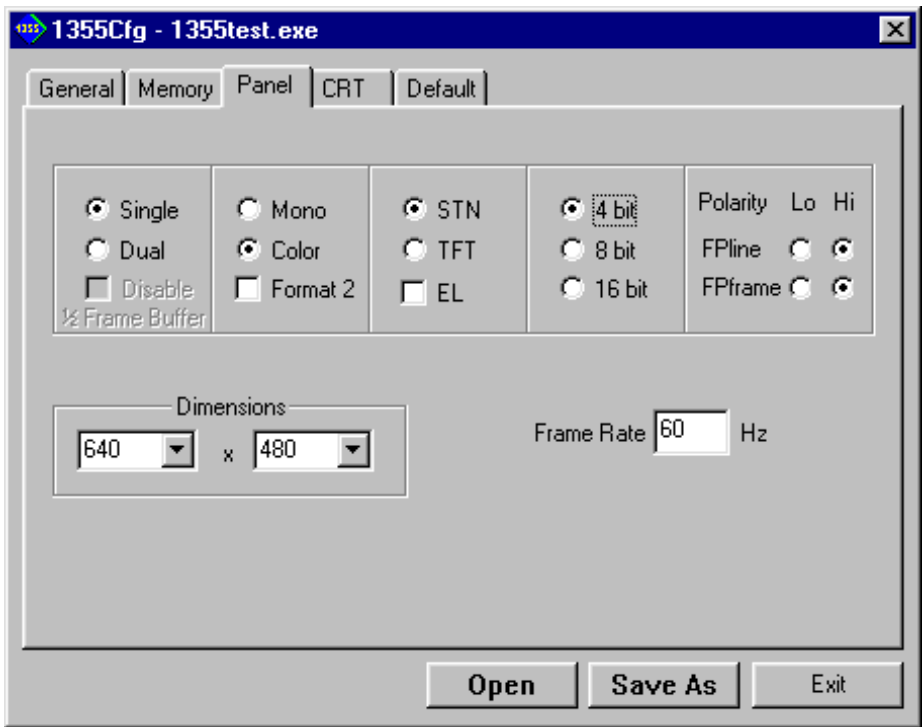


Figure 1-3 Panel Page

The Panel Page allows the user to select the following settings:

Panel Page	
Single/Dual	Select between a single and dual panel. If no panel exists, select single.
Disable half frame buffer	The half frame buffer is used only for dual panels. Disabling the half frame buffer is not recommended as this will reduce the display quality.
Mono/Color	Select between a monochrome and color panel. If no panel exists, select color.
Format 2	Select color passive LCD panel format 2. See the "SID13505 Hardware Functional Specification" for format 1/format 2 description.
STN/TFT	Select between a passive LCD and TFT/D-TFD panel.
EL	Enable EL panel support.
Panel Interface	Select panel interface width in bits. The bit width values will change when selecting between STN and TFT/D-TFD panels.
FPLine Polarity	Select the polarity of the FPLINE pulse.
FPframe Polarity	Select the polarity of the FPFRAME pulse.
Dimensions	Select the width and height of the panel in pixels.
Frame Rate	Select the desired frame rate.

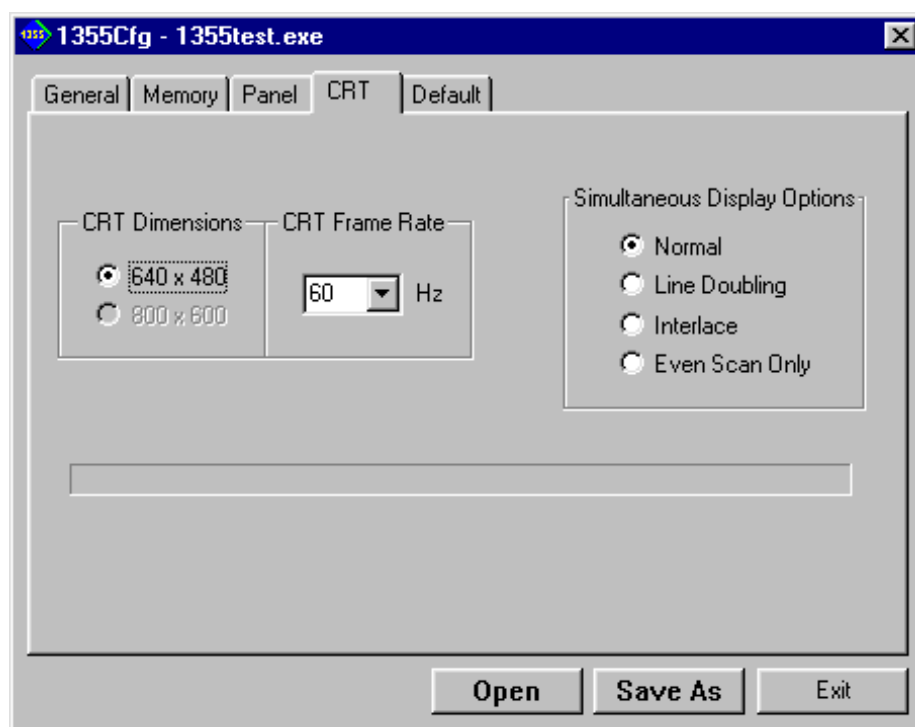
CRT Page

Figure 1-4 CRT Page

The CRT Page allows the user to select the following settings:

CRT Page	
CRT Dimensions	Select the desired resolution. See “Comments” on page 11 if the desired CRT dimensions are grayed out.
CRT Frame Rate	Select the desired frame rate. See “Comments” on page 11 to determine valid CRT frame rates.
Simultaneous Display Options	For simultaneous display only. Will be grayed out if simultaneous display is not supported based on the other configuration settings. For summary of Simultaneous Display options see the Hardware Functional Specification.

Default Page

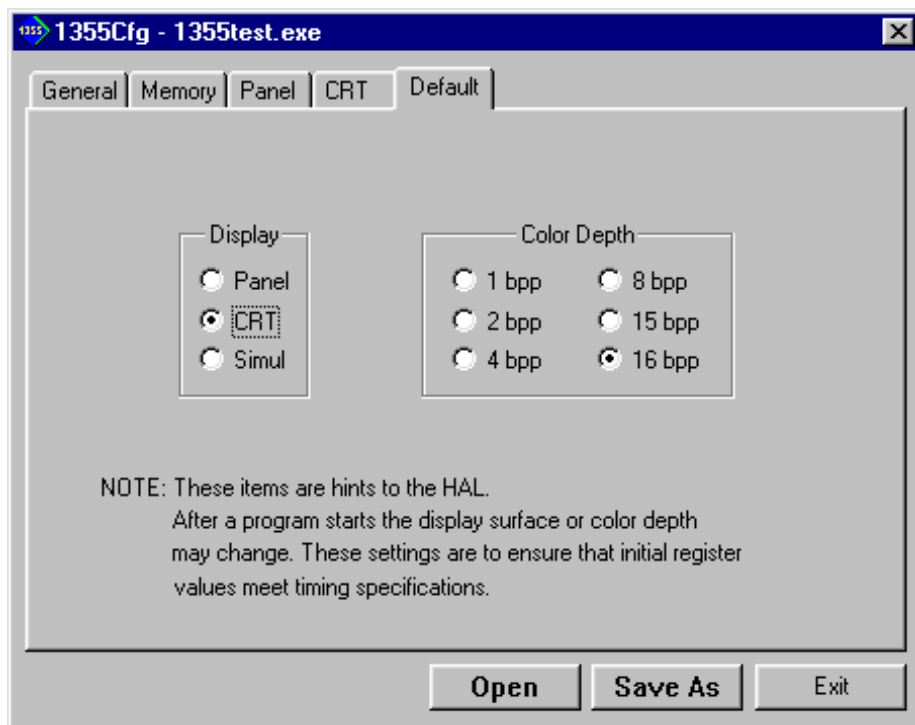
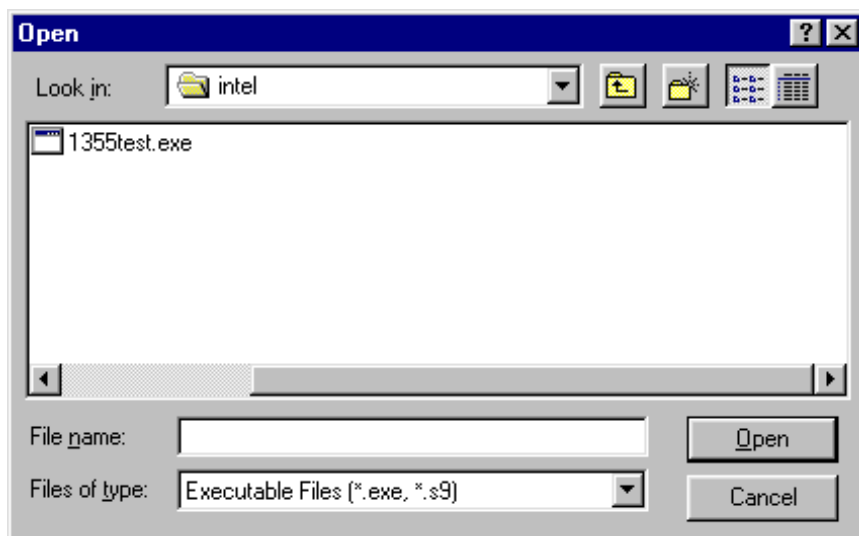


Figure 1-5 Default Page

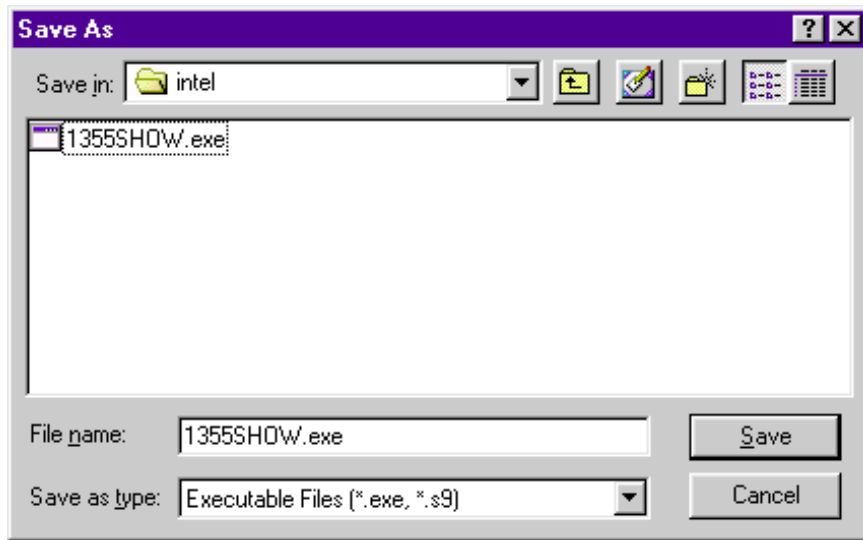
The Default Page allows the user to select the following settings:

Default Page	
Display	Select the default display device. Three display modes (LCD, CRT, and Simultaneous) are saved, but the S1D13505 software initializes the registers based on the default mode.
Color Depth	Select the default color depth.

Open Dialog Box



When the “OPEN” button is pressed on the main window, the Open Dialog Box is shown. 13505CFG will read the configuration values from a specific EXE file for Intel platforms, and from a specific S9 file for non-Intel platforms. The file must have been compiled using a valid version of the 13505HAL library.

Save As Dialog Box

When the “Save As” button is pressed on the main window, 13505CFG checks for any invalid configuration values and shows any appropriate warning or error messages. If it is possible to save the values, the Save As Dialog Box is shown.

The configuration values can be saved to a specific EXE file for Intel platforms, and to a specific S9 file for non-Intel platforms. The file must have been compiled using a valid version of the 13505HAL library. The configuration values can also be saved to an ASCII header file (ie. 13505reg.h) for use by the software/hardware developer.

Example

Configure for an 8-bit color single passive 640x480 LCD panel and the S5U13505P00C Evaluation Board on a PC:

General Page	
Register Address	0xE00000
Memory Address	0xC00000
CPU Bus Width	16 bit
ClkI	25175 kHz
Bus Clk	8000 kHz
Memory Page	
Timing (ns)	60 ns
Memory Type	EDO
WE# Control	2-CAS#
Refresh time (ms)	32 ms
Trc	104 ns
Trp	40 ns
Trac	60 ns
Suspend Mode Refresh	CAS before RAS
Panel Page	
Single/Dual	Single
Disable half frame buffer	(unchecked)
Mono/Color	Color
Format 2	(unchecked)
STN/TFT	STN
EL	(unchecked)
Panel Interface	8 bit
FPline Polarity	Hi
FPframe Polarity	Hi
Dimensions	640 x 480
Frame Rate	60
CRT Page	
CRT Dimensions	640 x 480
CRT Frame Rate	60
Simultaneous Display Options	Normal
Default Page	
Display	Panel
Color Depth	16 bpp

Note: The above configuration also supports simultaneous display and CRT only modes.

Comments

- It is assumed that the 13505CFG user is familiar with S1D13505 hardware and software. Refer to the “*S1D13505 Functional Hardware Specification*” and the “*S1D13505 Programming Notes and Examples*” for more details.
- 13505CFG verifies that the given configuration meets the limitations in the hardware specification. Part of this verification process is as follows:
 - The divide ratio for the *source clock/MClk* is determined based on *Table 14-3: Example Frame Rates with Ink Disabled* from the Functional Hardware Specification. According to this table, MClk cannot exceed 40 MHz for 50ns EDO-DRAM, MClk cannot exceed 33 MHz for 60ns EDO-DRAM, and MClk cannot exceed 25 MHz for 60ns FPM-DRAM. If MClk exceeds the maximum value, the MCLK value is set to the source clock divided by two ($MClk = source\ clock / 2$). Otherwise the MCLK value is set to the source clock ($MClk = source\ clock$). 13505CFG shows the MClk value on the General Page.
 - The divide ratio for $MClk/PCLK$ is determined based on *Table 14-1: Maximum PCLK Frequency with EDO-DRAM* and *Table 14-2: Maximum PCLK Frequency with FPM-DRAM*. Once this ratio is determined, $PCLK = MClk / ratio$. Note that there are two PClk divide ratios based on the three display modes: panel, CRT, and simultaneous display (CRT and simultaneous display use the same ratio). 13505CFG shows the PClk values for these three modes on the General Page.
 - The HNBP and VNBP values are calculated based on the desired frame rate for each of the three modes (panel, CRT, simultaneous), the display’s HDP (X resolution), VDP (Y resolution), and maximum PCLK as calculated in step 3.

$$FrameRate = \frac{PCLK}{(HDP + HNBP) \times (VDP + VNBP)}$$

- If it is not possible to reach the desired frame rate within 5%, an error message is shown when saving the configuration.
- When configuring either the CRT or TFT/D-TFD panel, the PClk must be the same as the required VESA frequency for the given VESA mode. The following VESA modes are supported:

Resolution	Frame Rate (Hz)	PCLK (MHz)	Supported DRAM Types
640x480	60	25.175	50ns EDO, 60ns EDO, 70ns EDO, 60ns FPM
	72	31.500	50ns EDO, 60ns EDO
	75	31.500	50ns EDO, 60ns EDO
	85	36.000	50ns EDO
800x600	56	36.000	50ns EDO
	60	40.000	50ns EDO

- 13505CFG does not support 50ns FPM-DRAM.
- 13505CFG programs TFT/D-TFD panels with the same VESA timings as a CRT, so the CRT restrictions shown in the hardware specification also apply to TFT/D-TFD panels. Consequently for TFT/D-TFD panels, use the CRT frame rate and CRT PCLK as described above.
- For simultaneous display, select a CRT VESA mode, and use the CRT’s frame rate for the panel’s frame rate.

Sample Program Messages

ERROR: Panel frame rate must be greater than zero

Select an appropriate panel frame rate as recommended in the panel specifications.

ERROR: Unable to save current settings

Could not save configuration values. The reason why is generally given before this message is shown.

ERROR: Invalid clock frequency selected

The ClkI frequency is either too low or too high.

ERROR: Max. clock for 50ns FPM is unspecified

13505CFG does not support 50ns FPM-DRAM.

WARNING: Cannot set panel display mode

This message is shown if the configuration settings do not meet the hardware specifications, or if the desired frame rate cannot be reached within 5%. See “Comments” on page 11 for more information.

WARNING: Cannot set simul display mode

This message is shown if the configuration settings do not meet the hardware specifications, or if the desired frame rate cannot be reached within 5%. See “Comments” on page 11 for more information.

WARNING: Cannot set CRT display mode

This message is shown if the configuration settings do not meet the hardware specifications, or if the desired frame rate cannot be reached within 5%. See “Comments” on page 11 for more information.

-PClk too slow to support 640 x 480

This message is shown after the “Cannot set ??? display mode” message. See “Comments” on page 11 to adjust the PClk.

-PClk too slow to support 800 x 600

This message is shown after the “Cannot set ??? display mode” message. See “Comments” on page 11 to adjust PClk.

Notice: Invalid clock selected for VESA frequencies. The monitor may not sync!

This message is shown in the CRT Page when the PClk is not set to a standard VESA frequency. See “Comments” on page 11 to adjust the PClk.

ERROR: Unknown HAL version.

When reading from or writing to a S1D13505 utility, 13505CFG could not find the start of a valid configuration table.

ERROR: Unable to open <filename>

Possible cause - no HAL information

13505CFG could not find the HAL configuration table.

ERROR: encountered while reading .S9 file.

The S9 file is corrupted.

ERROR: while attempting to write .S9 file.

The S9 file is corrupted.

2 *13505SHOW DEMONSTRATION PROGRAM*

2.1 *13505SHOW*

13505SHOW is designed to demonstrate and test some of the S1D13505 display capabilities. The program can cycle through all the color depths and display a pattern showing all available colors, or the user can specify a color depth and display configuration.

The 13505SHOW demonstration program must be configured and/or compiled to work with your hardware platform. The program 13505CFG.EXE can be used to configure 13505SHOW. Consult the “*13505CFG Configuration Program (X23A-B-001-02)*” for more information on configuring S1D13505 utilities.

This software is designed to work in both embedded and personal computer (PC) environments. For the embedded environment, it is assumed that the system has a means of downloading software from the PC to the target platform. Typically this is done by serial communications, where the PC uses a terminal program to send control commands and information to the target processor. Alternatively, the PC can program an EPROM, which is then placed in the target platform. Some target platforms can also communicate with the PC via a parallel port connection, or an Ethernet connection.

S1D13505 Supported Evaluation Platforms

13505SHOW supports the following S1D13505 evaluation platforms:

- PC system with an Intel 80x86 processor.
- M68332BCC (Business Card Computer) board, revision B, with a Motorola MC68332 processor.
- M68EC000IDP (Integrated Development Platform) board, revision 3.0, with a Motorola M68EC000 processor.
- SH3-LCEVB board, revision B, with an Hitachi SH-3 HD6417780 processor.

Installation

- **PC platform:** copy the file 13505SHOW.EXE to a directory that is in the DOS path on your hard drive.
- **Embedded platform:** download the program 13505SHOW to the system.

Usage

PC platform: at the prompt, type

```
13505show [b=??] [/a] [/crt] [/g] [/lcd] [/noinit] [/p] [/read] [/s] [/?].
```

Embedded platform: execute **13505show** and at the prompt, type the command line argument.

Where:	b=??	starts 13505SHOW at a user specified bit-per-pixel (bpp) level, where ?? can be: 1, 2, 4, 8, 15, or 16
	/a	automatically cycles through all video modes
	/crt	displays the image on the CRT
	/g	shows grid on the image
	/lcd	displays the image on the LCD panel
	/noinit	bypasses register initialization
	/p	draws the image in portrait mode
	/read	after drawing the image, continually read from the screen (for testing purposes)
	/s	displays vertical stripe pattern
	/?	displays the help screen

Note: Pressing the ESC key will exit the program.

13505SHOW Examples

The 13505SHOW demonstration program is designed to both demonstrate and test some of the features of the S1D13505. Some examples follow showing how to use the program in both instances.

Using 13505SHOW For Demonstration

1. To show color patterns which must be manually stepped through all bit-per-pixel modes, type the following:
13505SHOW

The program will display 16 bit-per-pixel mode. Press any key to go to the next screen. The program will display 15 bit-per-pixel mode. Once all screens are shown the program exits. To exit the program immediately press ESC.

2. To show color patterns which automatically step through all bit-per-pixel modes, type the following:

```
13505SHOW /a
```

The program will display 16 bit-per-pixel mode. Each screen is shown for approximately 1 second, then the next screen is automatically shown. The program exits after the last screen is shown. To exit the program immediately press CTRL+BREAK.

3. To show a color pattern for a specific bit-per-pixel mode, type the following:

```
13505SHOW b=[mode]
```

where mode = 1, 2, 4, 8, 15, or 16.

The program will display the requested screen and then exit.

4. To show the color patterns in portrait mode, type the following:

13505SHOW /p

The program will display 16 bit-per-pixel mode. Press any key to go to the next screen. The program will next display 15 bit-per-pixel mode and then 8 bit-per-pixel mode. Since portrait mode is limited to 8, 15, and 16 bit-per-pixel mode the program exits. To exit the program immediately press ESC.

The “/p” switch can be used in combination with other command line switches.

5. To show solid vertical stripes, type the following:

13505SHOW /s

The program will display 16 bit-per-pixel mode. Press any key to go to the next screen. The program will display 15 bit-per-pixel mode. Once all screens are shown the program exits. To exit the program immediately press ESC.

The “/s” switch can be used in combination with other command line switches.

Using 13505SHOW For Testing

1. To show a test grid other the color pattern, type the following:

13505SHOW b=8 /g

The program will display the 8 bit-per-pixel color pattern overlayed with a white grid pattern and then exit. Note the grid is not aligned with the color pattern, therefore the color boxes will not match the grid boxes.

The “/g” switch can be used in combination with other command line switches.

2. To test background memory reads, type the following:

13505SHOW b=16 /read

The program will test screen reads. If there is a problem with memory access, the displayed pattern will appear different than when the “/read” switch is not used. If there is a problem, check the configuration parameters of 13505SHOW using the utility 13505CFG. See the 13505CFG Configuration Program (X23A-B-001-02) for more information.

The “/read” switch should be used in combination with the “b=” setting, otherwise the test will always start with the 16 bit-per-pixel screen. To exit the program after using “/read”, press ESC and wait for a couple of seconds (the keystroke is checked after reading a full screen).

Comments

- 13505SHOW cannot show a greater color depth than the display allows.
- Portrait mode is available only for 8, 15, and 16 bit-per-pixel.
- When using a PC with the S5U13505P00C evaluation board, the PC must not have more than 12M bytes of system memory.
- 13505SHOW uses the panel color setup to determine whether to display a mono or color image on both the panel and the CRT. When editing in 13505CFG with CRT enabled and panel disabled, select “Color” from the “Panel” dialog box if you want the CRT to show color.
- For simultaneous display, select both “/lcd” and “/crt”.
- If the “b=” option is not used, 13505SHOW will cycle through all available bit-per-pixel modes.

Program Messages

ERROR: Could not initialize device.

These messages generally mean that the given hardware/software setup violates the timing limitations described in the S1D13505 Hardware Functional Specification.

ERROR: Too many devices registered.

There are too many display devices attached to the HAL. The HAL currently supports only one device.

ERROR: Could not register S1D13505F00A device.

A S1D13505 device was not found at the configured addresses. Check the configuration address using the “13505CFG configuration program”.

ERROR: Did not find a 13505 device.

The HAL was unable to read the revision code register on the S1D13505. Ensure that the S1D13505 hardware is installed and that the hardware platform has been set up correctly.

ERROR: Continual screen read will not work with the /a switch.

The continual screen read function reads one screen indefinitely, so it is not possible to automatically cycle through the video modes.

WARNING: b= option used with /noinit, so bit-per-pixel and display memory will NOT be changed.

The b= option requests that registers be changed for a given bit-per-pixel mode, while the /noinit option requests the opposite. To resolve this contradiction 13505SHOW will not change either the registers or the display memory. Consequently “13505SHOW b=?? /noinit” is only useful for continually reading the display memory.

UNSUPPORTED MODE: Cannot show ?? bpp in portrait mode.

Only 8, 15, 16 bit-per-pixel modes are supported in portrait mode.

3 *13505SPLT DISPLAY UTILITY*

3.1 *13505SPLT*

13505SPLT demonstrates S1D13505 split screen capability by showing two different areas of display memory on the screen simultaneously. Screen 1 shows horizontal bars and Screen 2 shows vertical bars.

Screen 1 memory is located at the start of the display buffer. Screen 2 memory is located immediately after Screen 1 in the display buffer. On user input, or elapsed time, the line compare register value is changed to adjust the amount of area displayed on either screen. The result is a movement up or down of screen 2 on the display.

The 13505SPLT display utility must be configured and/or compiled to work with your hardware platform. The program 13505CFG.EXE can be used to configure 13505SPLT. Consult the “*13505CFG Configuration Program (X23A-B-001-02)*” for more information on configuring S1D13505 utilities.

This software is designed to work in both embedded and personal computer (PC) environments. For the embedded environment, it is assumed that the system has a means of downloading software from the PC to the target platform. Typically, this is done by serial communications. The PC uses a terminal program to send control commands and information to the target processor. Alternatively, the PC can program an EPROM, which is then placed in the target platform. Some target platforms can also communicate with the PC via a parallel port connection, or an Ethernet connection.

S1D13505 Supported Evaluation Platforms

13505SPLT supports the following S1D13505 evaluation platforms:

- PC system with an Intel 80x86 processor.
- M68332BCC (Business Card Computer) board, revision B, with a Motorola MC68332 processor.
- M68EC000IDP (Integrated Development Platform) board, revision 3.0, with a Motorola M68EC000 processor.
- SH3-LCEVB board, revision B, with an Hitachi SH-3 HD6417780 processor.

Installation

PC platform: copy the file 13505SPLT.EXE to a directory that is in the DOS path on your hard drive.

Embedded platform: download the program 13505SPLT to the system.

Usage

PC platform: at the prompt, type **13505spl`t`** [**/a**] [**/?**].

Embedded platform: execute **13505spl`t`** and at the prompt, type the command line argument.

Where:	no argument	enables manual split screen operation
	/a	enables automatic split screen operation
	/?	displays the help screen

The following keyboard commands are for navigation within the program.

Manual mode:	↑	moves Screen 2 up one line
	↓	moves Screen 2 down one line
	CTRL-↑	moves Screen 2 up several lines
	CTRL-↓	moves Screen 2 down several lines
	HOME	Screen 2 moved up as high as possible
	END	Screen 2 moved down as low as possible

Automatic and Manual modes:

b	changes the color depth (bit-per-pixel)
ESC	exits 13505SPLT

13505SPLT Example

1. Type “13505spl`t` /a” to automatically move the split screen.
2. Press “b” to change the bit-per-pixel value from 16 to 15 bit-per-pixel.
3. Repeat step 2 for the remaining bit-per-pixel color depths: 8, 4, 2, and 1.
4. Press <ESC> to exit the program.

Comments

- When using a PC with the S5U13505P00C evaluation board, the PC must not have more than 12M bytes of system memory.

Program Messages

ERROR: Did not find a 13505 device.

The HAL was unable to read the revision code register on the S1D13505. Ensure that the S1D13505 hardware is installed and that the hardware platform has been set up correctly.

ERROR: Too many devices registered.

There are too many display devices attached to the HAL. The HAL currently supports only one device.

ERROR: Could not register S1D13505F00A device.

A S1D13505 device was not found at the configured addresses. Check the configuration address using the 13505CFG configuration program.

ERROR: Could not set ?? bit-per-pixel display mode.

This message generally means that the given hardware/software setup violates the timing limitations described in the “*S1D13505 Hardware Functional Specification*”.

4 *13505VIRT DISPLAY UTILITY*

4.1 *13505VIRT*

13505VIRT demonstrates the virtual display capability of the S1D13505. A virtual display is where the image to be displayed is larger than the physical display device (CRT or LCD). 13505VIRT uses panning and scrolling to allow the display device to show a “window” into the entire image.

The 13505VIRT display utility must be configured and/or compiled to work with your hardware platform. The program 13505CFG.EXE can be used to configure 13505VIRT. Consult the “*13505CFG Configuration Program (X23A-B-001-02)*” for more information on configuring S1D13505 utilities.

This software is designed to work in both embedded and personal computer (PC) environments. For the embedded environment, it is assumed that the system has a means of downloading software from the PC to the target platform. Typically this is done by serial communications, where the PC uses a terminal program to send control commands and information to the target processor. Alternatively, the PC can program an EPROM, which is then placed in the target platform. Some target platforms can also communicate with the PC via a parallel port connection, or an Ethernet connection.

S1D13505 Supported Evaluation Platforms

13505VIRT has been tested with the following S1D13505 supported evaluation platforms:

- PC system with an Intel 80x86 processor.
- M68332BCC (Business Card Computer) board, revision B, with a Motorola MC68332 processor.
- M68EC000IDP (Integrated Development Platform) board, revision 3.0, with a Motorola M68EC000 processor.
- SH3-LCEVB board, revision B, with an Hitachi SH-3 HD6417780 processor.

Installation

PC platform: copy the file 13505VIRT.EXE to a directory that is in the DOS path on your hard drive.

Embedded platform: download the program 13505VIRT to the system.

Usage

PC platform: at the prompt, type **13505virt [w=??] [/a] [/?]**.

Embedded platform: execute **13505virt** and at the prompt, type the command line argument.

Where:	no argument	panning and scrolling is performed manually
	w=??	for manual mode, specifies the width of the virtual display which must be a multiple of 8 and less than 2048 (the default width is double the physical panel width); the maximum height is based on the display memory
	/a	panning and scrolling is performed automatically
	/?	displays the help screen

The following keyboard commands are for navigation within the program.

Manual mode:	↑	scrolls up
	↓	scrolls down
	←	pans to the left
	→	pans to the right
	CTRL-↑	scrolls up several lines
	CTRL-↓	scrolls down several lines
	CTRL-←	pans to the left several lines
	CTRL-→	pans to the right several lines
	HOME	moves the display screen so that the upper right corner of the virtual screen shows in the display
	END	moves the display screen so that the lower left corner of the virtual screen shows in the display

Automatic and Manual modes:

b	changes the color depth (bit-per-pixel)
ESC	exits 13505VIRT

13505VIRT Example

1. Type “13505virt /a” to automatically pan and scroll.
2. Press “b” to change the bit-per-pixel value from 16 to 15 bit-per-pixel.
3. Repeat step 2 for the following bit-per-pixel values:
16, 15, 8, 4, 2, and 1.
4. Press <ESC> to exit the program.

Comments

- When using a PC with the S5U13505P00C evaluation board, the PC must not have more than 12M bytes of system memory.

Program Messages

ERROR: Did not find a 13505 device.

The HAL was unable to read the revision code register on the S1D13505. Ensure that the S1D13505 hardware is installed and that the hardware platform has been set up correctly.

ERROR: Too many devices registered.

There are too many display devices attached to the HAL. The HAL currently supports only one device.

ERROR: Could not register S1D13505F00A device.

A S1D13505 device was not found at the configured addresses. Check the configuration address using the 13505CFG configuration program.

ERROR: Not enough display buffer memory for ?? bpp.

There was not enough memory for a virtual screen.

5 13505PLAY DIAGNOSTIC UTILITY

5.1 13505PLAY

13505PLAY is a diagnostic utility which allows the user to read/write to all the S1D13505 Registers, Look-Up Tables and Display Buffer. 13505PLAY is similar to the DOS DEBUG program; commands are received from the standard input device, and output is sent to the standard output device (console for Intel, terminal for embedded platforms). This utility requires the target platform to support standard IO (stdio).

13505PLAY commands can be entered interactively by a user, or be executed from a script file. Scripting is a powerful feature which allows command sequences to be used repeatedly without re-entry.

The 13505PLAY diagnostic utility must be configured and/or compiled to work with your hardware platform. The program 13505CFG.EXE can be used to configure 13505PLAY. “Consult the 13505CFG Configuration Program (X23A-B-001-02)” for more information on configuring S1D13505 utilities.

This software is designed to work in both embedded and personal computer (PC) environments. For the embedded environment, it is assumed that the system has a means of downloading software from the PC to the target platform. Typically this is done by serial communications, where the PC uses a terminal program to send control commands and information to the target processor. Alternatively, the PC can program an EPROM, which is then placed in the target platform. Some target platforms can also communicate with the PC via a parallel port connection, or an Ethernet connection.

S1D13505 Supported Evaluation Platforms

13505PLAY supports the following S1D13505 evaluation platforms:

- PC system with an Intel 80x86 processor.
- M68332BCC (Business Card Computer) board, revision B, with a Motorola MC68332 processor.
- M68EC000IDP (Integrated Development Platform) board, revision 3.0, with a Motorola M68EC000 processor.
- SH3-LCEVB board, revision B, with an Hitachi SH-3 HD6417780 processor.

Installation

PC platform: copy the file 13505PLAY.EXE to a directory that is in the DOS path on your hard drive.

Embedded platform: download the program 13505PLAY to the system.

Usage

PC platform: at the prompt, type **13505play** [/?].

Embedded platform: execute **13505play** and at the prompt, type the command line argument.

Where: /? displays program version information.

The following commands are valid within the 13505PLAY program.

- | | |
|-------------------------------------|--|
| b 8 16 | <ul style="list-style-type: none">- Sets the ISA bus to 8 or 16 bits.- Only sets up the PAL on the S5U13505P00C evaluation board. There is no readback capability.- Only supported on a S5U13505P00C evaluation board for the PC platform. Switch 1-1 on the evaluation board must be set to the same bus width as used with this command. |
| f [w] addr1 addr2 data . . . | <ul style="list-style-type: none">- Fills bytes or words [w] from address 1 to address 2 with the data specified.- Data can be multiple values (e.g. F 0 20 1 2 3 4 fills 0 to 0x20 with a repeating pattern of 1 2 3 4). |
| h [lines] | <ul style="list-style-type: none">- Halts after <i>lines</i> of display. This feature halts the display during long read operations to prevent data from scrolling off the display. Similar to the DOS MORE command.- Set to 0 to disable this feature. |
| i [LCD] [CRT] | <ul style="list-style-type: none">- Initializes the chip with the specified configuration. The configuration is embedded in the 13505PLAY utility and can be changed using the 13505CFG utility. See the “13505CFG Configuration Program (X23A-B-001-02)”, for instructions on changing the configuration.- If the output device is specified, the user can select LCD, CRT, or both devices. |
| l index [red green blue] | <ul style="list-style-type: none">- Reads/writes Look-Up Table (LUT) values.- Writes data to the LUT[index] when data is specified.- Reads the LUT[index] when the data is not specified. |
| la | <ul style="list-style-type: none">- Reads all LUT values. |
| m [bpp] | <ul style="list-style-type: none">- Reads current mode information.- Sets the color depth (bpp) if “bpp” is specified. |

p 1 0	<ul style="list-style-type: none"> - Set power mode (hardware suspend). 1 = set hardware suspend. 0 = reset hardware suspend. - This command is only supported on a S5U13505P00C evaluation board for the PC platform.
q	<ul style="list-style-type: none"> - Quits the 13505PLAY utility.
r[w] addr [count]	<ul style="list-style-type: none"> - Reads number of bytes or words [w] from the address specified by "addr". If "count" is not specified, then 16 bytes/words are read.
v	<ul style="list-style-type: none"> - Calculates the frame rate from VNDP count (PC platform only).
w[w] addr data . . .	<ul style="list-style-type: none"> - Writes bytes or words [w] of data to the address specified by "addr". - Data can be multiple values (e.g. W 0 1 2 3 4 writes the byte values 1 2 3 4 starting at address 0).
x index [data]	<ul style="list-style-type: none"> - Reads/writes the registers. - Writes data to REG[index] when "data" is specified. Reads data from REG[index] when "data" is not specified.
-	
xa	<ul style="list-style-type: none"> - Reads all registers.
?	<ul style="list-style-type: none"> - Displays Help information.

13505PLAY Example

1. Type "13505PLAY" to start the program.
2. Type "?" for help.
3. Type "i" to initialize the registers.
4. Type "xa" to display the contents of the registers.
5. Type "x 5" to read register 5.
6. Type "x 3 10" to write 10h to register 3.
7. Type "f 0 ffff aa" to fill the first FFFFh bytes of the display buffer with AAh.
8. Type "f 0 1ffff aa" to fill 2M bytes of the display buffer with AAh.
9. Type "r 0 100" to read the first 100h bytes of the display buffer.
10. Type "q" to exit the program.

Scripting

13505PLAY can be driven by a script file. This is useful when:

- there is no display output and a current register status is required.
- various registers must be quickly changed to view results.

A script file is an ASCII text file with one 13505PLAY command per line. All scripts must end with a “q” (quit) command.

On a PC platform, a typical script command line might be:

“13505PLAY < dumpregs.scr > results.”

This causes the file “dumpregs.scr” to be interpreted as commands by 13505PLAY and the results to be sent to the file “results.”

Example: Create an ASCII text file that contains the commands `i`, `xa`, and `q`.

; This file initializes the S1D13505 and reads the registers.

; Note: after a semicolon (;), all characters on a line are ignored.

; Note: all script files must end with the “q” command.

i

xa

q

Comments

- All numeric values are considered to be hexadecimal unless identified otherwise. For example, 10 = 10h = 16 decimal; 10t = 10 decimal; 010b = 2 decimal.
- Redirecting commands from a script file (PC platform) allows those commands to be executed as though they were typed.
- When using a PC with the S5U13505P00C evaluation board, the PC must not have more than 12M bytes of system memory.

Program Messages

WARNING: Did not find a 13505 device.

The HAL was unable to read the revision code register on the S1D13505. Ensure that the S1D13505 hardware is installed and that the hardware platform has been set up correctly.

ERROR: Failed to change to ?? mode.

Could not change to CRT, LCD, or SIMUL mode. This message generally means that the given hardware/software setup violates the timing limitations described in the S1D13505 Hardware Functional Specification.

ERROR: Could not change to ?? bit-per-pixel.

This message generally means that the given hardware/software setup violates the timing limitations described in the S1D13505 Hardware Functional Specification.

ERROR: Too many devices registered.

There are too many display devices attached to the HAL. The HAL currently supports only one device.

ERROR: Could not register S1D13505F00A device.

A S1D13505 device was not found at the configured addresses. Check the configuration address using the 13505CFG configuration program.

ERROR: Insufficient memory for ?? bit-per-pixel.

The given display resolution requires a larger display buffer than is available to store the image. Either increase the amount of display buffer or select a lower color depth (bpp).

WARNING: Clocks are too fast for given mode.

This message is only shown if the “m” command was entered and the MCLK/PCLK frequencies violated the timings in the “*S1D13505 Hardware Functional Specification*”.

6 13505BMP DEMONSTRATION PROGRAM

6.1 13505BMP

13505BMP is a demonstration utility used to show the S1D13505 display capabilities by rendering bitmap images on the display. The program will display any bitmap in Windows BMP file format and then exit. A 24-bit true color bitmap will be truncated to 16 bit-per-pixel mode.

13505BMP is designed to operate on a personal computer (PC) in the DOS environment. Other embedded platforms are not supported due to the lack of memory and structured file systems.

The 13505BMP demonstration utility must be configured and/or compiled to work with your hardware configuration. The program 13505CFG.EXE can be used to configure 13505BMP. “Consult the 13505CFG Configuration Program (X23A-B-001-02)” for more information on configuring S1D13505 utilities.

S1D13505 Supported Evaluation Platforms

13505BMP supports the following S1D13505 evaluation platforms:

- PC system with an Intel 80x86 processor.

Note: The 13505BMP source code may be modified by the OEM to support other evaluation platforms.

Installation

Copy the file 13505BMPEXE to a directory that is in the DOS path on your hard drive.

Usage

At the prompt, type **13505bmp bmpfile [/a] [/crt] [/lcd] [/p] [/?]**.

Where:	 bmpfile 	filename of a windows format bmp image
	 /a 	adds a 2 second delay before automatically exiting
	 /crt 	displays the image on a CRT
	 /lcd 	displays the image on a LCD
	 /p 	portrait mode
	 /? 	displays the Help screen

Note: 13505BMP will automatically finish execution and return to the prompt.

13505BMP Examples

To display a bmp image on a CRT, type the following:
13505BMP bmpfile.bmp /crt

To display a bmp image on a LCD, type the following:
13505BMP bmpfile.bmp /lcd

To display a bmp image on a LCD in portrait mode, type the following:
13505BMP bmpfile.bmp /lcd /p

To display a bmp image on a CRT and delay 2 seconds before exiting, type the following:
13505BMP bmpfile.bmp /crt /a

Comments

- 13505BMP displays only Windows BMP format images.
- The PC must not have more than 12M bytes of memory when used with the S5U13505P00C evaluation board.
- Only the green component of the image will be seen on a monochrome display.

Program Messages

ERROR: Could not initialize device.

These messages generally mean that the given hardware/software setup violates the timing limitations described in the “*S1D13505 Hardware Functional Specification*”.

ERROR: Did not find a 13505 device.

The HAL was unable to read the revision code register on the S1D13505. Ensure that the S1D13505 hardware is installed and that the hardware platform has been set up correctly.

ERROR: Too many devices registered.

There are too many display devices attached to the HAL. The HAL currently supports only one device.

ERROR: Could not register S1D13505F00A device.

A S1D13505 device was not found at the configured addresses. Check the configuration address using the 13505CFG configuration program.

ERROR: Did not detect S1D13505.

The HAL was unable to read the revision code register on the S1D13505. Ensure that the S1D13505 hardware is installed and that the hardware platform has been set up correctly.

ERROR: Insufficient memory for ?? bit-per-pixel.

The given display resolution requires more memory than is available to store one complete image. Either increase the amount of display memory or select an image with a lower bit-per-pixel value.

7 *13505PWR SOFTWARE SUSPEND POWER SEQUENCING UTILITY*

7.1 *13505PWR*

13505PWR is a diagnostic utility used to test some of the power save capabilities of the S1D13505. 13505PWR enables or disables the software suspend mode, hardware suspend mode, and the LCD, allowing testing of the power sequencing in each mode.

To measure the timing for power sequencing, GPIO pin 1 is used to trigger an oscilloscope at the point the requested power sequencing function is activated/deactivated. For further information on LCD Power Sequencing and Power Save Modes, refer to the “*S1D13505 Programming Notes and Examples*” and the “*S1D13505 Functional Hardware Specification*”.

The 13505PWR software suspend power sequencing utility must be configured and/or compiled to work with your hardware platform. The program 13505CFG.EXE can be used to configure 13505PWR. Consult the “*13505CFG Configuration Program (X23A-B-001-02)*” for more information on configuring S1D13505 utilities.

This software is designed to work in both embedded and personal computer (PC) environments. For the embedded environment, it is assumed that the system has a means of downloading software from the PC to the target platform. Typically this is done by serial communications, where the PC uses a terminal program to send control commands and information to the target processor. Alternatively, the PC can program an EPROM, which is then placed in the target platform. Some target platforms can also communicate with the PC via a parallel port connection, or an Ethernet connection.

S1D13505 Supported Evaluation Platforms

13505PWR supports the following S1D13505 evaluation platforms:

- PC system with an Intel 80x86 processor.
- M68332BCC (Business Card Computer) board, revision B, with a Motorola MC68332 processor.
- M68EC000IDP (Integrated Development Platform) board, revision 3.0, with a Motorola M68EC000 processor.
- SH3-LCEVB board, revision B, with an Hitachi SH-3 HD6417780 processor.

Installation

PC platform: copy the file 13505PWR.EXE to a directory that is in the DOS path on your hard drive.

Embedded platform: download the program 13505PWR to the system.

Usage

PC platform: at the prompt, type

```
13505pwr [/software | /hardware | /lcd] [/enable | /disable] [/i]
[/0 | /1] [/?].
```

Embedded platform: execute **13505pwr** and at the prompt, type the command line argument.

Where:	/software	selects software suspend
	/hardware	selects hardware suspend (PC only)
	/lcd	selects the LCD
	/enable	activates software suspend, hardware suspend, or the LCD
	/disable	deactivates software suspend, hardware suspend, or the LCD
	/i	initializes registers
	/0	GPIO1 triggers on falling edge (1->0)
	/1	GPIO1 triggers on rising edge (0->1)
	/?	displays this usage message

Note: 13505PWR will automatically finish execution and return to the prompt.

13505PWR Examples

To enable software suspend mode, type the following:

```
13505PWR /software /enable
```

To disable software suspend mode, type the following:

```
13505PWR /software /disable
```

To enable hardware suspend mode, type the following:

```
13505PWR /hardware /enable
```

To disable hardware suspend mode, type the following:

```
13505PWR /hardware /disable
```

To enable the LCD, type the following:

```
13505PWR /lcd /enable
```

To disable the LCD, type the following:

```
13505PWR /lcd /disable
```

Comments

- The /i argument is to be used when the registers have not been previously initialized.
- When using a PC with the S5U13505P00C evaluation board, the PC must not have more than 12M bytes of system memory.
- GPIO1 is used to signal when the software suspend mode, hardware suspend mode, or LCD has been enabled or disabled.
- Hardware suspend is changed by reading or writing to a memory address decoded by the PAL on the S5U13505P00C evaluation board. This PAL is currently only used for PC platforms, so the S5U13505P00C evaluation board does not support hardware suspend on embedded platforms.

Program Messages

ERROR: Did not detect S1D13505.

The HAL was unable to read the revision code register on the S1D13505. Ensure that the S1D13505 hardware is installed and that the hardware platform has been set up correctly.

ERROR: Unknown command line argument.

An invalid command line argument was entered. Enter a valid command line argument.

ERROR: Already selected SOFTWARE.

Command line argument `/software` was selected more than once. Select `/software` only once.

ERROR: Already selected HARDWARE.

Command line argument `/hardware` was selected more than once. Select `/hardware` only once.

ERROR: Already selected LCD.

Command line argument `/lcd` was selected more than once. Select `/lcd` only once.

ERROR: Already selected ENABLE.

Command line argument `/enable` was selected more than once. Select `/enable` only once.

ERROR: Already selected DISABLE.

Command line argument `/disable` was selected more than once. Select `/disable` only once.

ERROR: Select `/software`, `/hardware` or `/lcd`.

Did not select one of the following command line arguments: `/software`, `/hardware` or `/lcd`. Select `/software`, `/hardware` or `/lcd`.

ERROR: Select `/enable` or `/disable`.

Neither command line argument `/enable` or `/disable` was selected. Select `/enable` or `/disable`.

ERROR: Too many devices registered.

There are too many display devices attached to the HAL. The HAL currently supports only one device.

ERROR: Could not register S1D13505F00A device.

A S1D13505 device was not found at the configured addresses. Check the configuration address using the 13505CFG configuration program.

S1D13505F00A
Embedded RAMDAC LCD/CRT Controller

S5U13505P00C
ISA Bus
Evaluation Board
User's Manual

Table of Contents

1	INTRODUCTION	4-1
1.1	Features	4-1
2	INSTALLATION AND CONFIGURATION.....	4-2
3	LCD INTERFACE PIN MAPPING	4-3
4	CPU/BUS INTERFACE CONNECTOR PINOUTS.....	4-4
5	HOST BUS INTERFACE PIN MAPPING	4-6
6	TECHNICAL DESCRIPTION.....	4-7
6.1	ISA Bus Support.....	4-7
6.2	Non-ISA Bus Support.....	4-7
6.3	DRAM Support	4-7
6.4	Decode Logic	4-7
6.5	Clock Input Support.....	4-7
6.6	Monochrome LCD Panel Support	4-8
6.7	Color Passive LCD Panel Support	4-8
6.8	Color TFT/D-TFD LCD Panel Support.....	4-8
6.9	CRT Support	4-8
6.10	Power Save Modes	4-8
6.11	Adjustable LCD Panel Negative Power Supply	4-8
6.12	Adjustable LCD Panel Positive Power Supply	4-8
6.13	CPU/Bus Interface Header Strips.....	4-9
6.14	Schematic Notes	4-9
7	PARTS LIST	4-10
8	SCHEMATIC DIAGRAMS	4-11

List of Figures

Figure 8-1	S5U13505P00C Schematic Diagram (1 of 4)	4-11
Figure 8-2	S5U13505P00C Schematic Diagram (2 of 4)	4-12
Figure 8-3	S5U13505P00C Schematic Diagram (3 of 4)	4-13
Figure 8-4	S5U13505P00C Schematic Diagram (4 of 4)	4-14

List of Tables

Table 2-1	Configuration DIP Switch Settings	4-2
Table 2-2	Host Bus Selection.....	4-2
Table 2-3	Jumper Settings	4-2
Table 3-1	LCD Signal Connector (J6)	4-3
Table 4-1	CPU/BUS Connector (H1) Pinout	4-4
Table 4-2	CPU/BUS Connector (H2) Pinout	4-5
Table 5-1	CPU Interface Pin Mapping	4-6

1 INTRODUCTION

This manual describes the setup and operation of the S5U13505P00C Rev. 1.0 Evaluation Board. Implemented using the S1D13505 Embedded RAMDAC LCD/CRT Controller, the S5U13505P00C is designed for the ISA bus environment. It also provides CPU/Bus interface connectors for non-ISA bus support.

For more information regarding the S1D13505, refer to the “*S1D13505 Hardware Functional Specification*”.

1.1 Features

- 128-pin QFP15 surface mount package.
- SMT technology for all appropriate devices.
- 4/8-bit monochrome passive LCD panel support.
- 4/8/16-bit color passive LCD panel support.
- 9/12/18-bit LCD TFT/D-TFD panel support.
- Embedded RAMDAC for CRT support.
- 16-bit ISA bus support.
- Oscillator support for CLKI (up to 40.0MHz).
- 5.0V 1M x 16 EDO-DRAM (2M byte).
- Support for software and hardware suspend modes.
- On-board adjustable LCD bias power supply (+24..38V or -24..14V).
- CPU/Bus interface header strips for non-ISA bus support.

2 INSTALLATION AND CONFIGURATION

The S1D13505 has 16 configuration inputs MD[15:0] which are read on the rising edge of RESET#. Inputs MD[5:1] are fully configurable on this evaluation board for different host bus selections; one eight-position DIP switch is provided for this purpose. All remaining configuration inputs are hard-wired. See the “*S1D13505 Hardware Functional Specification*” for more information.

The following settings are recommended when using the S5U13505P00C with the ISA bus.

Table 2-1 Configuration DIP Switch Settings

Switch	Signal	Closed (1)	Open (0)
SW1-1	MD1	See “Host Bus Selection” table below	See “Host Bus Selection” table below
SW1-2	MD2		
SW1-3	MD3		
SW1-4	MD4	Little Endian	Big Endian
SW1-5	MD5	Wait# signal is active high	Wait# signal is active low
SW1-6	MD13	Reserved	
SW1-7	MD14		
SW1-8	MD15		

Table 2-2 Host Bus Selection

MD3 / SW1-3	MD2 / SW1-2	MD1 / SW1-1	Host Bus Interface
open (0)	open (0)	open (0)	SH-3/SH-4 bus interface
open (0)	open (0)	closed (1)	MC68K bus 1 interface (e.g. MC68000)
open (0)	closed (1)	open (0)	MC68K bus 2 interface (e.g. MC68030)
open (0)	closed (1)	closed (1)	Generic bus interface
closed (1)	open (0)	open (0)	Reserved
closed (1)	open (0)	closed (1)	MIPS/ISA
closed (1)	closed (1)	open (0)	PowerPC
closed (1)	closed (1)	closed (1)	PC Card (PCMCIA)

= recommended settings (configured for ISA bus support)

Table 2-3 Jumper Settings

	Description	1-2	2-3
JP1	DRDY (pin 76, S1D13505)	Pin 76 connected to J6 pin 38	Pin 76 connected to J6 pin 35
JP2	LCD VDD selection	5.0V LCD driver VDD	3.3V LCD driver VDD

Note: JP1 is for internal use only, default setting is 1-2.

3 LCD INTERFACE PIN MAPPING

Table 3-1 LCD Signal Connector (J6)

S1D13505 Pin Names	Connector Pin No.	Color TFT/D-TFD			Color Passive			Mono Passive	
		9-bit	12-bit	18-bit	4-bit	8-bit	16-bit	4-bit	8-bit
FPDAT0	1	R2	R3	R5		LD0	LD0		LD0
FPDAT1	3	R1	R2	R4		LD1	LD1		LD1
FPDAT2	5	R0	R1	R3		LD2	LD2		LD2
FPDAT3	7	G2	G3	G5		LD3	LD3		LD3
FPDAT4	9	G1	G2	G4	UD0	UD0	UD0	UD0	UD0
FPDAT5	11	G0	G1	G3	UD1	UD1	UD1	UD1	UD1
FPDAT6	13	B2	B3	B5	UD2	UD2	UD2	UD2	UD2
FPDAT7	15	B1	B2	B4	UD3	UD3	UD3	UD3	UD3
FPDAT8	17	B0	B1	B3			LD4		
FPDAT9	19		R0	R2			LD5		
FPDAT10	21			R1			LD6		
FPDAT11	23		G0	G2			LD7		
FPDAT12	25			G1			UD4		
FPDAT13	27			G0			UD5		
FPDAT14	29		B0	B2			UD6		
FPDAT15	31			B1			UD7		
FPSHIFT	33	FPSHIFT							
DRDY	35					FPSHIFT2			
FPLINE	37	FPLINE							
FPFRAME	39	FPFRAME							
GND	2-26 (Even Pins)	GND							
N/C	28								
VEEH	30	Adjustable -24..-14V negative LCD bias							
LCDVCC	32	Jumper selectable +3.3V/+5V							
+12V	34	+12V							
VDDH	36	Adjustable +15..+38V positive LCD bias							
DRDY	38	DRDY			MOD	FPSHIFT2	MOD		
LCDPWR#	40	LCDPWR#							

4 CPU/BUS INTERFACE CONNECTOR PINOUTS

Table 4-1 CPU/BUS Connector (H1) Pinout

Connector Pin No.	Comments
1	Connected to DB0 of the S1D13505
2	Connected to DB1 of the S1D13505
3	Connected to DB2 of the S1D13505
4	Connected to DB3 of the S1D13505
5	Ground
6	Ground
7	Connected to DB4 of the S1D13505
8	Connected to DB5 of the S1D13505
9	Connected to DB6 of the S1D13505
10	Connected to DB7 of the S1D13505
11	Ground
12	Ground
13	Connected to DB8 of the S1D13505
14	Connected to DB9 of the S1D13505
15	Connected to DB10 of the S1D13505
16	Connected to DB11 of the S1D13505
17	Ground
18	Ground
19	Connected to DB12 of the S1D13505
20	Connected to DB13 of the S1D13505
21	Connected to DB14 of the S1D13505
22	Connected to DB15 of the S1D13505
23	Connected to RESET# of the S1D13505
24	Ground
25	Ground
26	Ground
27	+12 volt supply
28	+12 volt supply
29	Connected to WE0# of the S1D13505
30	Connected to WAIT# of the S1D13505
31	Connected to CS# of the S1D13505
32	Connected to MR# of the S1D13505
33	Connected to WE1# of the S1D13505
34	Not connected

Table 4-2 CPU/BUS Connector (H2) Pinout

Connector Pin No.	Comments
1	Connected to AB0 of the S1D13505
2	Connected to AB1 of the S1D13505
3	Connected to AB2 of the S1D13505
4	Connected to AB3 of the S1D13505
5	Connected to AB4 of the S1D13505
6	Connected to AB5 of the S1D13505
7	Connected to AB6 of the S1D13505
8	Connected to AB7 of the S1D13505
9	Ground
10	Ground
11	Connected to AB8 of the S1D13505
12	Connected to AB9 of the S1D13505
13	Connected to AB10 of the S1D13505
14	Connected to AB11 of the S1D13505
15	Connected to AB12 of the S1D13505
16	Connected to AB13 of the S1D13505
17	Ground
18	Ground
19	Connected to AB14 of the S1D13505
20	Connected to AB15 of the S1D13505
21	Connected to AB16 of the S1D13505
22	Connected to AB17 of the S1D13505
23	Connected to AB18 of the S1D13505
24	Connected to AB19 of the S1D13505
25	Ground
26	Ground
27	+5 volt supply
28	+5 volt supply
29	Connected to RD/WR# of the S1D13505
30	Connected to BS# of the S1D13505
31	Connected to BUSCLK of the S1D13505
32	Connected to RD# of the S1D13505
33	Connected to AB20 of the S1D13505
34	Not connected

5 *Host Bus Interface Pin Mapping*

Table 5-1 CPU Interface Pin Mapping

S1D13505 Pin Names	SH-3	SH-4	MC68K Bus 1	MC68K Bus 2	Generic	MIPS/ISA	PowerPC	PCMCIA
AB20	A20	A20	A20	A20	A20	LatchA20	A11	A20
AB[16:13]	A[19:13]	A[19:13]	A[19:13]	A[19:13]	A[19:13]	SA[19:13]	A[12:18]	A[19:13]
AB[12:1]	A[12:1]	A[12:1]	A[12:1]	A[12:1]	A[12:1]	SA[12:1]	A[19:30]	A[12:1]
AB0	A0	A0	LDS#	A0	A0	SA0	A31	A0
DB[15:0]	D[15:0]	D[15:0]	D[15:0]	D[31:16]	D[15:0]	SD[15:0]	D[0:15]	D[15:0]
WE1#	WE1#	WE1#	UDS#	DS#	WE1#	SBHE#	BI#	-CE2
M/R#	External Decode							
CS#	External Decode							
BUSCLK	CKIO	CKIO	CLK	CLK	BCLK	CLK	CLKOUT	CLKI
BS#	BS#	BS#	AS#	AS#	VDD	VDD	TS#	VDD
RD/WR#	RD/WR#	RD/WR#	R/W#	R/W#	RD1#	VDD	RD/WR#	-CE1
RD#	RD#	RD#	VDD	SIZ1	RD0#	MEMR#	TSIZ0	-OE
WE0#	WE0#	WE0#	VDD	SIZ0	WE0#	MEMW#	TSIZ1	-WE
WAIT#	WAIT#	RDY	DTACK#	DSACK1#	WAIT#	IOCHRDY	TA#	-WAIT
RESET#	RESET#	RESET#	RESET#	RESET#	RESET#	inverted RESET	RESET#	inverted RESET

6 TECHNICAL DESCRIPTION

6.1 ISA Bus Support

The S5U13505P00C directly supports the 16-bit ISA bus environment. All the configuration options [MD15:0] are either hard-wired or selectable through the eight-position DIP Switch S1. Refer to Table 2-1, “Configuration DIP Switch Settings,” on page 4-2 for detail.

- Notes:**
1. The S5U13505P00C evaluation board supports a 16-bit ISA bus only.
 2. The S1D13505 is a memory-mapped device with 2M bytes of linear addressed display buffer and a separate 47 byte register space. On the S5U13505P00C, the S1D13505 2M byte display buffer has been mapped to a start address of C00000h and the registers have been mapped to a start address of E00000h.
 3. When using this board in a PC environment, system memory must be limited to 12M bytes, to prevent the system addresses will conflict with the S1D13505 display buffer/register addresses.

6.2 Non-ISA Bus Support

This evaluation board is specifically designed to support the standard 16-bit ISA bus. However, the S1D13505 directly supports many other host bus interfaces. Header strips H1 and H2 have been provided and contain all the necessary I/O pins to interface to these buses. See, Section 4, “CPU/Bus Interface Connector Pinouts”, Table 2-1, “Configuration DIP Switch Settings,” and Table 2-3, “Jumper Settings,” for details.

When using the header strips to provide the bus interface observe the following:

- All I/O signals on the ISA bus card edge must be isolated from the ISA bus (do not plug the card into a computer). Voltage lines are provided on the header strips.
- For the ISA bus, a 22V10 PAL (U4, socketed) is currently used to provide the S1D13505 CS# (pin 4), M/R# (pin 5) and other decode logic signals. This functionality must now be provided externally. Remove the PAL from its socket to eliminate conflicts resulting from two different outputs driving the same input. Refer to Table 2-2, “Host Bus Selection,” for connection details.

6.3 DRAM Support

The S1D13505 supports 256K x 16 as well as 1M x 16 FPM/EDO-DRAM in symmetrical and asymmetrical formats.

The S5U13505P00C board supports a 5.0V 1M x 16 symmetrical EDO-DRAM (42-pin SOJ package). This provides a 2M byte display buffer.

6.4 Decode Logic

This board utilizes the MIPS/ISA Interface of the S1D13505 (see the “S1D13505 Hardware Functional Specification”).

All required decode logic is provided through a 22V10 PAL (U4, socketed).

6.5 Clock Input Support

The S1D13505 supports up to a 40.0MHz input clock frequency. A 40.0MHz oscillator (U2, socketed) is provided on the S5U13505P00C board as the clock (CLKI) source.

6.6 Monochrome LCD Panel Support

The S1D13505 supports 4 and 8-bit, dual and single, monochrome passive LCD panels. All necessary signals are provided on the 40-pin ribbon cable header J6. The interface signals on the cable are alternated with grounds to reduce crosstalk and noise.

Refer to Table 3-1, “LCD Signal Connector (J6),” for connection information.

6.7 Color Passive LCD Panel Support

The S1D13505 directly supports 4, 8 and 16-bit, dual and single, color passive LCD panels. All the necessary signals are provided on the 40-pin ribbon cable header J6. The interface signals on the cable are alternated with grounds to reduce crosstalk and noise.

Refer to Table 3-1, “LCD Signal Connector (J6),” for connection information.

6.8 Color TFT/D-TFD LCD Panel Support

The S1D13505 supports 9, 12 and 18-bit active matrix color TFT/D-TFD panels. All the necessary signals can also be found on the 40-pin LCD connector J6. The interface signals on the cable are alternated with grounds to reduce crosstalk and noise.

When supporting an 18-bit TFT/D-TFD panel, the S1D13505 can display 64K of a possible 256K colors. A maximum 16 of the possible 18 bits of LCD data are available from the S1D13505. Refer to the “*S1D13505 Hardware Functional Specification*” for details.

Refer to Table 3-1, “LCD Signal Connector (J6),” for connection information.

6.9 CRT Support

This evaluation board provides CRT support through the S1D13505’s embedded RAMDAC. Refer to the “*S1D13505 Hardware Functional Specification*” for details.

6.10 Power Save Modes

The S1D13505 supports one hardware suspend and one software suspend Power Save Mode.

6.11 Adjustable LCD Panel Negative Power Supply

Most monochrome passive LCD panels require a negative power supply to provide between -18V and -23V ($I_{out}=45mA$). For ease of implementation, such a power supply has been provided as an integral part of this design. The signal VLCD can be adjusted by R29 to supply an output voltage from -14V to -23V and is enabled/disabled by the S1D13505 control signal LCDPWR#.

Determine the panel’s specific power requirements and set the potentiometer accordingly before connecting the panel.

6.12 Adjustable LCD Panel Positive Power Supply

Most passive LCD passive color panels and most single monochrome 640x480 passive LCD panels require a positive power supply to provide between +23V and +40V ($I_{out}=45mA$). For ease of implementation, such a power supply has been provided as an integral part of this design. The signal VDDH can be adjusted by R23 to provide an output voltage from +23V to +40V and is enabled/disabled by the S1D13505 control signal LCDPWR#.

Determine the panel’s specific power requirements and set the potentiometer accordingly before connecting the panel.

6.13 CPU/Bus Interface Header Strips

All of the CPU/Bus interface pins of the S1D13505 are connected to the header strips H1 and H2 for easy interface to a CPU, or bus other than ISA.

Refer to Table 4-1, “CPU/BUS Connector (H1) Pinout,” and Table 4-2, “CPU/BUS Connector (H2) Pinout,” for specific settings.

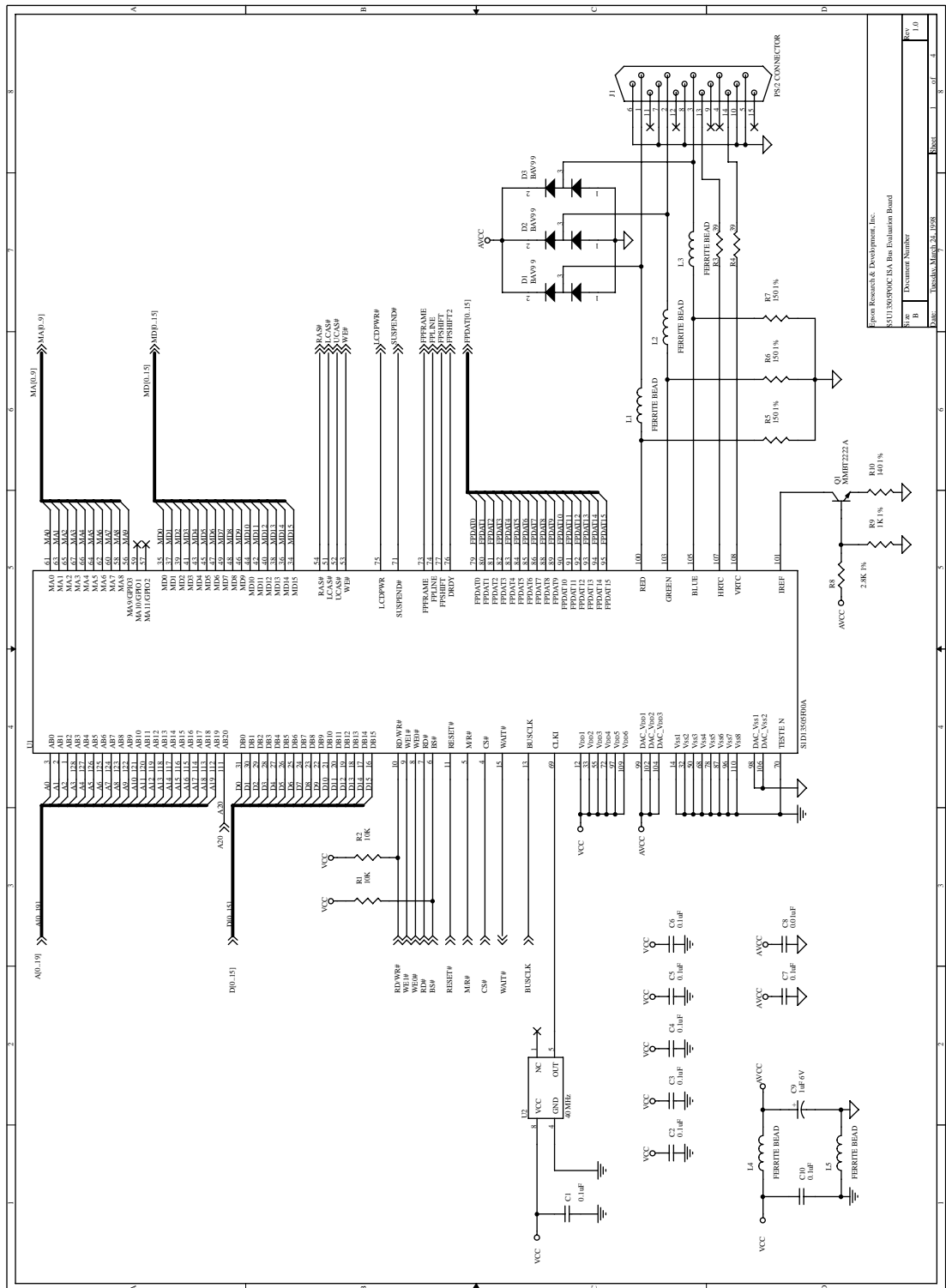
Note: These headers only provide the CPU/Bus interface signals from the S1D13505. When another host bus interface is selected through [MD3:1] configuration, appropriate external decode logic **MUST** be used to access the S1D13505. See the section “Host Bus Interface Pin Mapping” of the S1D13505 Hardware Functional Specification.

6.14 Schematic Notes

The following schematics are for reference only and may not reflect actual implementation. Please request updated information before starting any hardware design.

7 PARTS LIST

Item No.	Qty/Board	Designation	Part Value	Description
1	16	C1,C2,C3,C4,C5,C6,C7,C10,C11, C12,C13,C18,C25,C27,C28,C29	0.1uF	0805 ceramic capacitor
2	1	C8	0.01uF	0805 ceramic capacitor
3	2	C9,C30	1uF 6V	Tantalum capacitor size A
4	2	C14,C19	47uF 6V	Tantalum capacitor size D
5	3	C15,C16,C17	4.7uF 50V	Tantalum capacitor size D
6	1	C20	56uF 35V	Low-ESR electrolytic
7	4	C21,C22,C23,C24	4.7uF 16V	Tantalum capacitor size B
9	3	D1,D2,D3	BAV99	Signal diode
10	2	H1,H2	HEADER 17X2	
11	2	JP1,JP2	HEADER 3	
12	1	J1	VGA connector	
13	1	J2	AT CON-A	
14	1	J3	AT CON-B	
15	1	J4	AT CON-C	
16	1	J5	AT CON-D	
17	1	J6	CON40A	
18	6	L1,L2,L3,L4,L5,L7	Ferrite bead	Philips BDS3/3/8.9-4S2
19	1	L6	Inductor 1uH	
20	2	Q1,Q3	MMBT2222A	
21	1	Q2	MMBT2907A	
22	10	R1,R2,R21,R26,R30,R31,R32,R33, R34,R35	10K	0805 resistor
23	2	R3,R4	39 Ohms	0805 resistor
24	3	R5,R6,R7	150 1%	0805 resistor
25	1	R8	2.8K 1%	0805 resistor
26	1	R9	1K 1%	0805 resistor
27	1	R10	140 1%	0805 resistor
28	10	R11,R12,R13,R14,R15,R16,R17, R18,R19,R20	15K	0805 resistor
29	1	R22	470K	0805 resistor
30	1	R23	200K Pot.	
31	1	R24	14K	0805 resistor
32	1	R25	4.7K	0805 resistor
33	2	R28,R27	100K	0805 resistor
34	1	R29	100K Pot.	
35	1	S1	SW DIP-8	
36	1	U1	S1D13505F00A	
37	1	U2	40MHz oscillator	
38	1	U3	MT4C1M16E5DJS-5	50ns self-refresh EDO DRAM
39	1	U4	PAL22V10-15	
40	1	U5	RD-0412	Xentek RD-0412
41	1	U6	EPN001	Xentek EPN001
42	3	U7,U8,U9	74AHC244	
43	1	U10	LT1117CM-3.3	"5V to 3.3V regulator, 800mA"



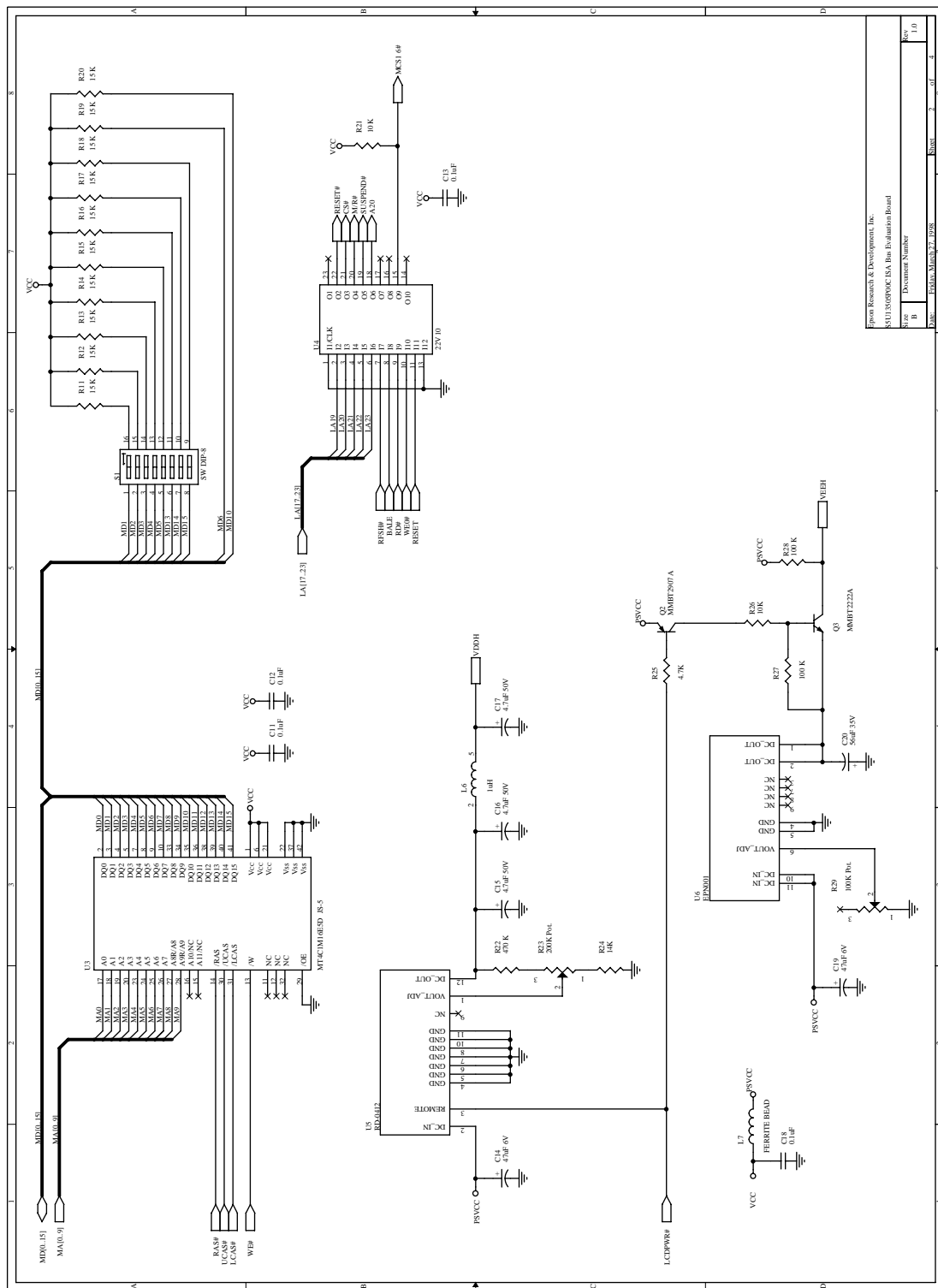
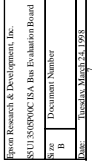


Figure 8-2 S5U13505P00C Schematic Diagram (2 of 4)



S5U13505P00C REV. 1.0 ISA BUS EVALUATION BOARD EPSON
USER'S MANUAL (X23A-G-004-04)

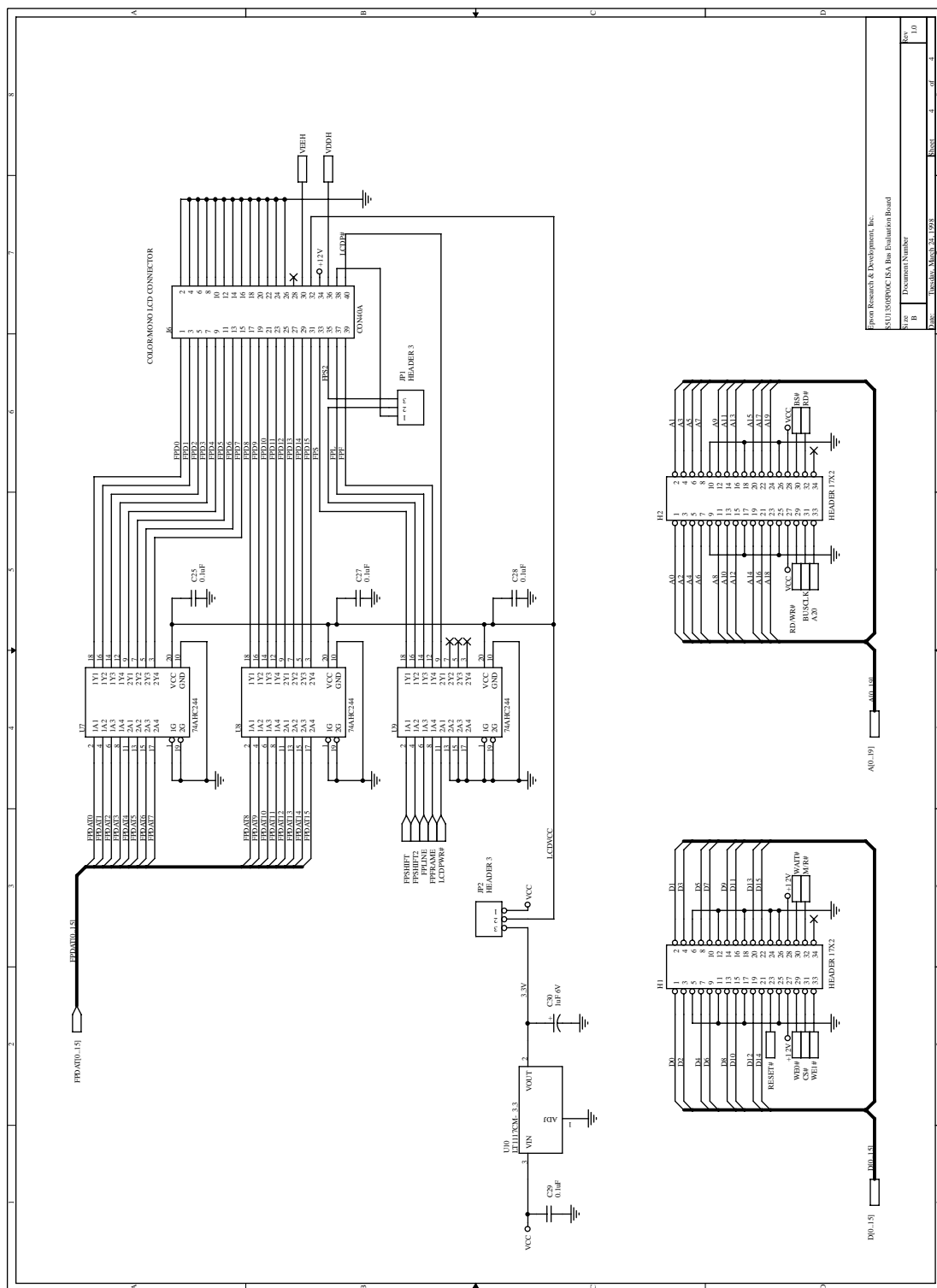


Figure 8-4 S5U13505P00C Schematic Diagram (4 of 4)

S1D13505F00A

Embedded RAMDAC LCD/CRT Controller

Application Notes

Table of Contents

1	INTERFACING THE S1D13505 TO THE PC CARD BUS	5-1
1.1	Introduction.....	5-1
	General Description	5-1
	Register/Memory Mapping	5-2
1.2	S1D13505 Configuration	5-3
	Hardware Configuration	5-3
	Performance	5-3
2	INTERFACING TO THE NEC Vr4102™ /Vr4111™ MICROPROCESSOR	5-4
2.1	Introduction.....	5-4
	General Description	5-4
2.2	S1D13505 Configuration	5-5
	Hardware Description	5-5
	NEC Vr4102™/Vr4111™ Configuration.....	5-5
3	INTERFACING TO THE NEC Vr4121™ MICROPROCESSOR	5-6
3.1	Introduction.....	5-6
3.2	Configuration	5-7
	Hardware Description	5-7
	NEC Vr4121™ Configuration.....	5-8
	Memory Mapping and Aliasing.....	5-8
	S1D13505 Pin Mapping.....	5-9
	S1D13505 Configuration.....	5-9
4	INTERFACING TO THE PHILIPS MIPS PR31500/PR31700 PROCESSOR	5-10
4.1	Introduction.....	5-10
4.2	Interfacing to the PR31500/PR31700.....	5-10
4.3	S1D13505 Host Bus Interface.....	5-11
	PR31500/PR31700 Host Bus Interface Pin Mapping	5-11
	PR31500/PR31700 Host Bus Interface Signals.....	5-11
4.4	Direct Connection to the Philips PR31500/PR31700	5-12
	Hardware Description	5-12
	S1D13505 Configuration.....	5-14
	Memory Mapping and Aliasing.....	5-14
4.5	System Design Using the IT8368E PC Card Buffer	5-15
	Hardware Description	5-15
	IT8368E Configuration	5-15
	S1D13505 Configuration.....	5-15
4.6	Software	5-16
5	INTERFACING TO THE TOSHIBA MIPS TX3912 PROCESSOR	5-17
5.1	Introduction.....	5-17
5.2	Interfacing to the TX3912	5-17
5.3	S1D13505 Host Bus Interface.....	5-18
	TX3912 Host Bus Interface Pin Mapping.....	5-18
	TX3912 Host Bus Interface Signals	5-19
5.4	Direct Connection to the Toshiba TX3912	5-19
	Hardware Description	5-19
	S1D13505 Configuration.....	5-21
	Memory Mapping and Aliasing.....	5-21
5.5	System Design Using the IT8368E PC Card Buffer	5-22
	Hardware Description	5-22
	IT8368E Configuration	5-22
	S1D13505 Configuration.....	5-22
5.6	Software	5-23

6	INTERFACING TO THE MOTOROLA MPC821 MICROPROCESSOR.....	5-24
6.1	Introduction	5-24
	General Description	5-24
6.2	Hardware Connections.....	5-25
6.3	Hardware Configuration	5-27
	S1D13505 Configuration	5-27
	MPC821 Chip Select Configuration	5-28
6.4	Test Software	5-29
	Test Software Source Code.....	5-29
7	DAC APPLICATION NOTES	5-30
7.1	DAC Application Notes.....	5-30
	Introduction	5-30
	Notes When Operating the S1D13505 Built-in DAC.....	5-30
	DAC Isolated Power Source	5-31
	Peripheral Circuit of DAC Pins.....	5-32
	RED/GREEN/BLUE Pins	5-33
	IREF Pin.....	5-34
	HRTC and VRTC Pins	5-34
	Notes When the DAC is Not Used.....	5-35
	Isolated DAC Power Pin	5-35
	Isolated DAC Signal Pins.....	5-36
8	POWER CONSUMPTION	5-37
8.1	S1D13505 Power Consumption.....	5-37
	Conditions.....	5-38
8.2	Summary.....	5-38

List of Figures

Figure 1-1	Schematic for S1D13505 in PC Card Bus.....	5-2
Figure 2-1	NEC VR4102™ to S1D13505 Configuration Schematic	5-4
Figure 3-1	NEC VR4121™ to S1D13505 Configuration Schematic	5-7
Figure 4-1	Typical Implementation of Direct Connection.....	5-13
Figure 4-2	IT8368E Implementation Block Diagram.....	5-15
Figure 5-1	Typical Implementation of Direct Connection.....	5-20
Figure 5-2	IT8368E Implementation Block Diagram.....	5-22
Figure 6-1	Schematic for MPC821/S1D13505 Interface	5-25

List of Tables

Table 1-1	Register/Memory Mapping Typical Implementation	5-2
Table 1-2	Summary of Power On/Reset Options	5-3
Table 2-1	Summary of Power-On/Reset Options	5-5
Table 3-1	S1D13505 to NEC VR4121™ Pin Mapping.....	5-9
Table 3-2	S1D13505 Configuration for NEC VR4121™	5-9
Table 4-1	PR31500/PR31700 Host Bus Interface Pin Mapping.....	5-11
Table 4-2	S1D13505 Configuration for Direct Connection	5-14
Table 4-3	PR31500/PR31700 to PC Card Slots Address Remapping for Direct Connection	5-14
Table 5-1	TX3912 Host Bus Interface Pin Mapping	5-18
Table 5-2	S1D13505 Configuration for Direct Connection	5-21
Table 5-3	TX3912 to PC Card Slots Address Remapping for Direct Connection.....	5-21
Table 6-1	List of Connections from MPC821ADS to S1D13505	5-26
Table 6-2	S1D13505 Configuration Settings	5-27
Table 6-3	Host Bus Selection	5-27
Table 6-4	Memory Configuration	5-27
Table 7-1	S1D13505 Power Supply Pin Description	5-31
Table 7-2	S1D13505 DAC Pin Description.....	5-32
Table 8-1	S1D13505 Total Power Consumption	5-38

1 INTERFACING THE S1D13505 TO THE PC CARD BUS

1.1 Introduction

This application note describes the hardware necessary to provide an interface between the S1D13505 Embedded RAMDAC LCD/CRT Controller and the PC Card (PCMCIA) bus.

For further information on the S1D13505 please refer to its “*Hardware Functional Specification*”. For information on the PC Card standard contact the Personal Computer Memory Card International Association (PCMCIA).

General Description

The S1D13505 was designed to directly support a variety of CPU's, providing an interface to their unique 'local bus'. However, in order to provide support for processors not having an appropriate local bus, the S1D13505 also supports both a Generic (ISA-Bus like interface) and a specific PC Card (PCMCIA) interface.

The S1D13505 provides a direct 'glueless' interface to the 16-bit PC Card bus, with the following exceptions;

1. The RESET# signal on the S1D13505 is active low and therefore must be inverted to support the active high RESET provided by the PC Card interface
2. Although the S1D13505 supports an asynchronous bus interface, a clock source is required on the BCLK input pin.

Note: The BCLK frequency is not critical and does not have to be synchronized to the bus signals.

The following diagram demonstrates a typical implementation of the interface.

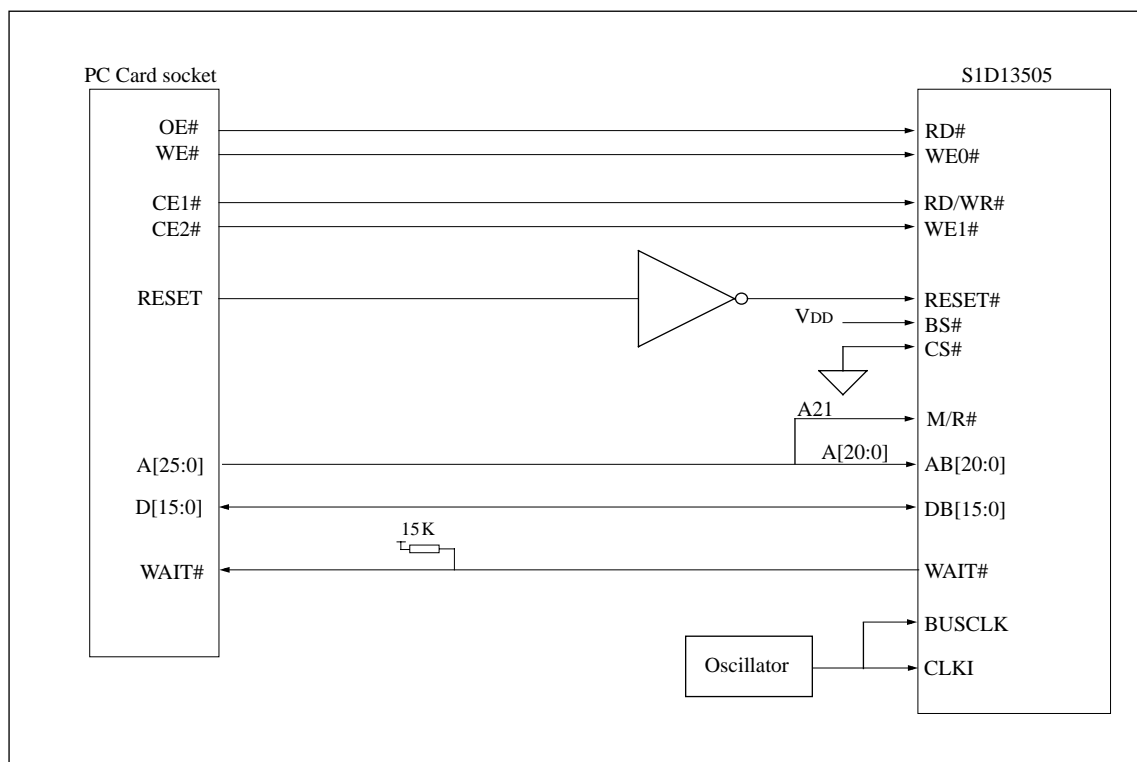


Figure 1-1 Schematic for S1D13505 in PC Card Bus

Register/Memory Mapping

The PC Card socket provides 64M byte of address space. The S1D13505 is a memory mapped device; 2M byte Display Buffer, and 47 byte Internal Register Set.

Table 1-1 shows a typical implementation having the Chip Select pin (CS#) connected to ground (always enabled) and the Memory/Register Select pin (M/R#) connected to Address bit A21. This provides the following decoding:

Table 1-1 Register/Memory Mapping Typical Implementation

CS#	M/R# (A21)	Address Range	Function
0	0	0 - 1FFFFFFh	Internal Register Set decoded ¹
0	1	200000h - 3FFFFFFh	Display Buffer decode

Note: The internal register set only requires 47 bytes. Therefore, without further resolution on the decode select logic (M/R# connected to A21), the entire register set is aliased for every 64 byte boundary within the specified address range above. Since address bits A25 to A22 are ignored, the S1D13505 registers and display memory are aliased sixteen times. If aliasing is not desirable, the upper addresses must be fully decoded.

1.2 S1D13505 Configuration

Hardware Configuration

The S1D13505 is configured on power-up by latching the power-on state of the DRAM data pins, MD[15:0]. Refer to the “*S1D13505 Hardware Functional Specification*” for details.

The “partial” table below shows those configuration settings important in interfacing to the PC Card bus as shaded.

Table 1-2 Summary of Power On/Reset Options

S1D13505 Pin Name	value on this pin at rising edge of RESET# is used to configure:(1/0)	
	1	0
MD0	8-bit host bus interface	16-bit host bus interface
MD[3:1]	111 = PC Card bus interface	
MD4	Little Endian	Big Endian
MD5	WAIT# is active high (1 = insert wait state)	WAIT# is active low (0 = insert wait state)
MD11	Alternate host bus interface selected	Primary host bus interface selected
MD12	BUSCLK input divided by two	BUSCLK input not divided by two

Performance

The S1D13505 PC Card Interface is specified to support a BCLK of up to 50MHz, and therefore can provide a high performance display solution.

2 INTERFACING TO THE NEC VR4102™ /VR4111™ MICROPROCESSOR

2.1 Introduction

This application note describes the hardware and software environment necessary to provide an interface between the S1D13505 Embedded RAMDAC LCD/CRT Controller and the NEC VR4102™ (μPD30102) or VR4111™ (μPD30111) Microprocessor.

For further information on either device refer to their respective technical specifications.

General Description

The NEC VR4102™ Microprocessor is specifically designed to support an external LCD controller. It provides the necessary internal address decoding and control signals. The S1D13505 can make use of the interface to easier support the NEC VR4102™.

The diagram below shows a typical implementation.

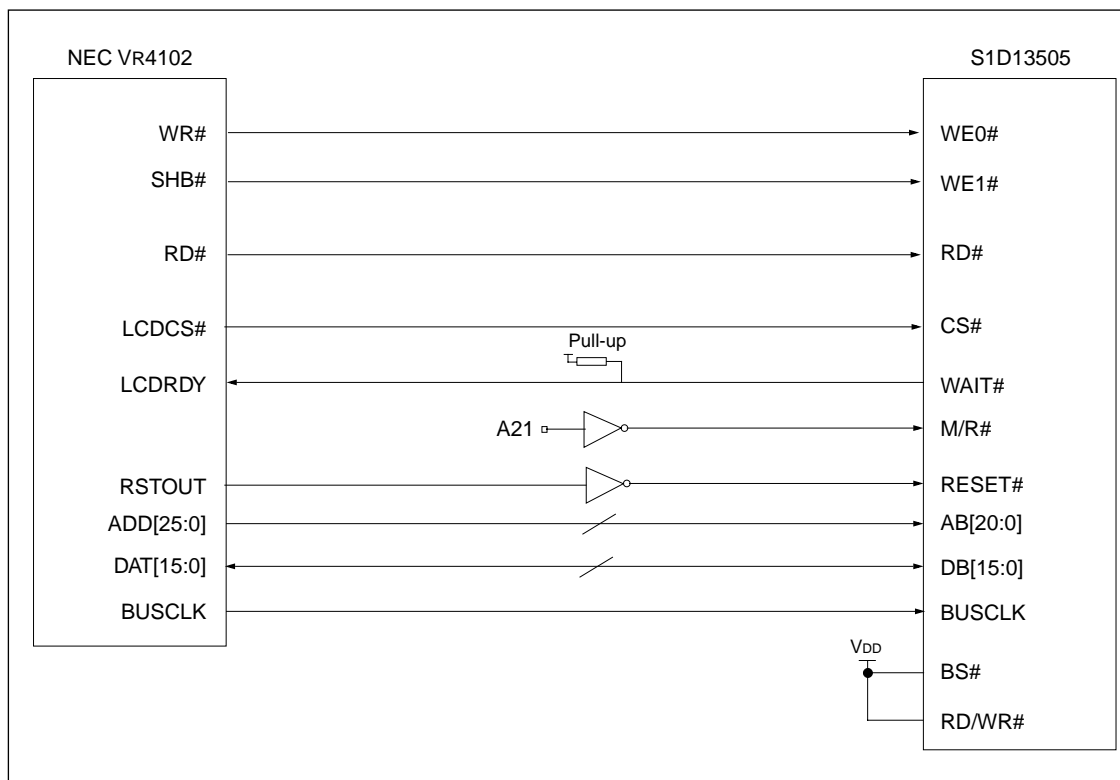


Figure 2-1 NEC VR4102™ to S1D13505 Configuration Schematic

2.2 S1D13505 Configuration

Hardware Description

The S1D13505 is configured on power-up by latching the power-on state of the DRAM data pins, MD[15:0]. Refer to the “*S1D13505 Hardware Specification*” for details.

The “partial” table below shows those configuration settings important to the NEC VR4102™/VR4111™ CPU interface.

Table 2-1 Summary of Power-On/Reset Options

S1D13505 Pin Name	Value on this pin at rising edge of RESET# is used to configure: (1/0)	
	1	0
MD0	8-bit host bus interface	16-bit host bus interface
MD[3:1]	101 = MIPS/ISA bus interface	
MD4	Little Endian	Big Endian
MD5	WAIT# is active high (1 = insert wait state)	WAIT# is active low (0 = insert wait state)
MD11	Alternate host bus interface selected	Primary host bus interface selected

NEC VR4102™/VR4111™ Configuration

The NEC VR4102™/VR4111™ provides the internal address decoding necessary to map to an external LCD controller. Physical address 0x0A000000h to 0x0FFFFFFh (16M bytes) is reserved for an external LCD controller.

The S1D13505 supports up to 2M bytes of display buffer. The NEC VR4102™/VR4111™ address line A21 is used to select between the S1D13505 display buffer and internal registers.

3 INTERFACING TO THE NEC VR4121™ MICROPROCESSOR

3.1 Introduction

This application note describes the hardware and software environment necessary to provide an interface between the SID13505 Embedded RAMDAC LCD/CRT Controller and the NEC VR4121™ (μPD30121) microprocessor.

For further information on the SID13505, refer to the “*SID13505 Hardware Functional Specification*”.

3.2 Configuration

Hardware Description

The NEC Vr4121™ microprocessor is specifically designed to support an external LCD controller. It provides all the necessary internal address decoding and control signals required by the S1D13505.

The diagram below shows a typical implementation utilizing the S1D13505.

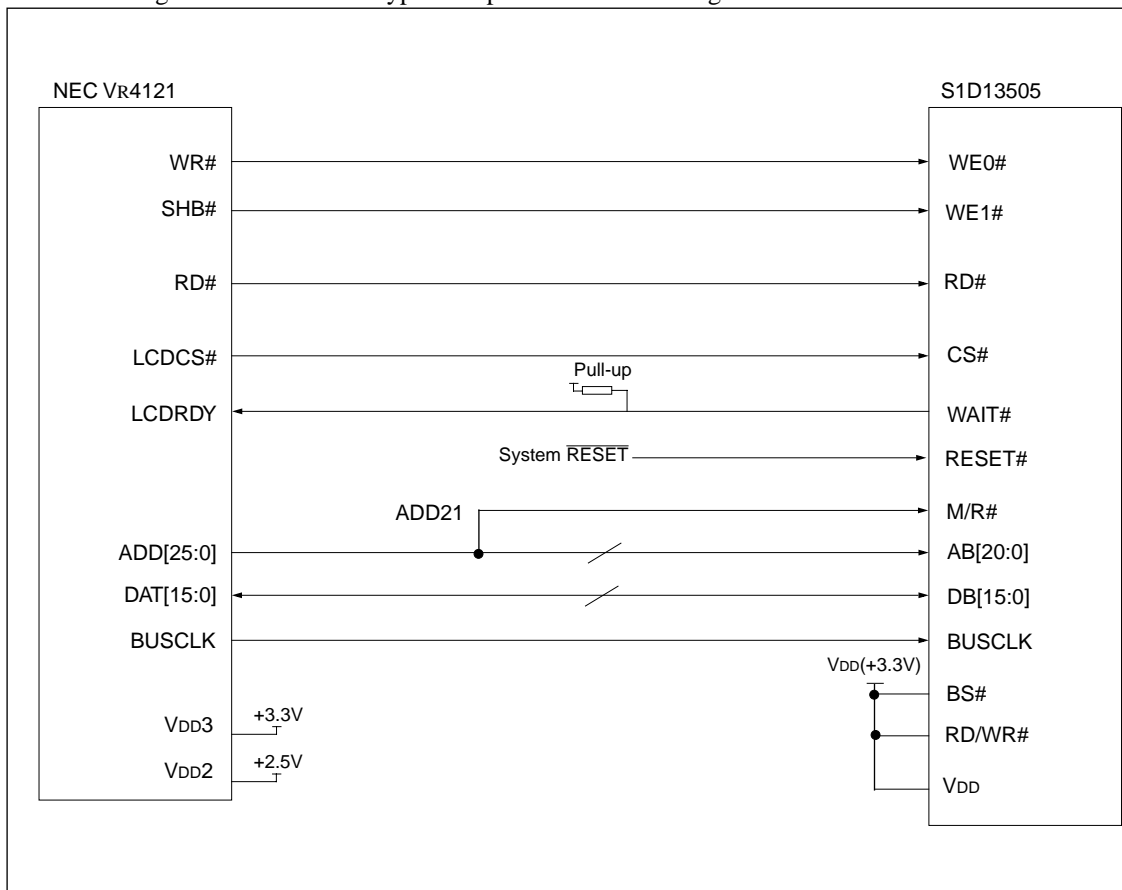


Figure 3-1 NEC Vr4121™ to S1D13505 Configuration Schematic

NEC VR4121™ Configuration

The NEC VR4121™ register BCUCNTREG1 bit ISAM/LCD must be set to “0”. A “0” indicates that the reserved address space is for the LCD controller, and not for the high-speed ISA memory. The register BCUCNTREG2 bit GMODE must be set to “1” to indicate that a non-inverting data bus is used for LCD controller accesses.

The LCD interface must be set to operate using a 16-bit data bus. This is accomplished by setting the NEC VR4121™ register BCUCNTREG3 bit LCD32/ISA32 to “0”.

Note: Setting the register BCUCNTREG3 bit LCD32/ISA32 to “0” affects both the LCD controller and high-speed ISA memory access.

The frequency of the BUSCLK output can be programmed from the state of pins TxD/CLKSEL2, RTS#/CLKSEL1 and DTR#/CLKSEL0 during reset, and from the PMU (Power Management Unit) configuration registers of the NEC VR4121™. The S1D13505 works at any of the frequencies provided by the NEC VR4121™.

Memory Mapping and Aliasing

The NEC VR4121™ provides the internal address decoding required by an external LCD controller. The physical address range from 0A000000h to 0AFFFFFFh (16M bytes) is reserved for use by an external LCD controller (e.g. S1D13505).

The S1D13505 supports up to 2M bytes of display buffer. The NEC VR4121™ address line ADD21 (connected to M/R#) is used to select between the S1D13505 display buffer (ADD21=1) and the S1D13505 internal registers (ADD21=0). NEC VR4121™ address lines ADD[23:22] are ignored, thus the S1D13505 is aliased four times at 4M byte intervals over the LCD controller address range. Address lines ADD[25:24] are set at 10b and never change while the LCD controller is being addressed.

S1D13505 Pin Mapping

Table 3-1 S1D13505 to NEC VR4121™ Pin Mapping

S1D13505 Pin Name	NEC VR4121 Pin Name
WE1#	SHB#
RD#	RD#
WE0#	WR#
WAIT#	LCDRDY
CS#	LCDCS#
DB[15:0]	DAT[15:0]
AB[20:0]	ADD[20:0]
M/R#	ADD21
RESET#	$\overline{\text{RESET}}$
BUSCLK	BUSCLK
BS#, RD/WR# are connected to VDD (+3.3V)	

S1D13505 Configuration

The S1D13505 latches MD0 through MD15 to allow selection of the bus mode and other configuration data on the rising edge of RESET#. The partial table below shows those configuration settings relevant to the MIPS/ISA host bus interface used by the NEC VR4121™ microprocessor.

Table 3-2 S1D13505 Configuration for NEC VR4121™

S1D13505 Pin Name	Value on this pin at rising edge of RESET# is used to configure: (1/0)	
	1	0
MD0	8-bit host bus interface	16-bit host bus interface
MD[3:1]	101 = MIPS/ISA host bus interface	
MD4	Little Endian	Big Endian
MD5	WAIT# is active high (1 = insert wait state)	WAIT# is active low (0 = insert wait state)
MD11	Alternate host bus interface selected	Primary host bus interface selected

 = configuration for NEC VR4121™ microprocessor

4 INTERFACING TO THE PHILIPS MIPS PR31500/PR31700 PROCESSOR

4.1 Introduction

This application note describes the hardware and software environment necessary to provide an interface between the S1D13505 Embedded RAMDAC LCD/CRT Controller and the Philips MIPS PR31500/PR31700 Processor.

4.2 Interfacing to the PR31500/PR31700

The Philips MIPS PR31500/PR31700 processor supports up to two PC Card (PCMCIA) slots. It is through this host bus interface that the S1D13505 connects to the PR31500/PR31700 processor.

The S1D13505 can be successfully interfaced using one of the following configurations:

- Direct connection to the PR31500/PR31700.
- System design using the ITE IT8368E PC Card/GPIO buffer chip.

4.3 S1D13505 Host Bus Interface

The S1D13505 implements a 16-bit host bus interface specifically for interfacing to the PR31500/PR31700 microprocessor.

The PR31500/PR31700 host bus interface is selected by the S1D13505 on the rising edge of RESET#. After releasing reset, the bus interface signals assume their selected configuration. For details on S1D13505 configuration, see “S1D13505 Configuration” on page 14.

Note: At reset, the Host Interface Disable bit in the Miscellaneous Disable Register (REG[1Bh] bit 7) is set to 1. This means that only REG[1Ah] (read-only) and REG[1Bh] are accessible **until a write to REG[1Bh] sets bit 7 to 0 making all registers accessible**. When debugging a new hardware design, this can sometimes give the appearance that the interface is not working, so it is important to remember to clear this bit before proceeding with debugging.

PR31500/PR31700 Host Bus Interface Pin Mapping

The following table shows the function of each host bus interface signal.

Table 4-1 PR31500/PR31700 Host Bus Interface Pin Mapping

S1D13505 Pin Name	Philips PR31500/PR31700
AB20	ALE
AB19	/CARDREG
AB18	/CARDIORD
AB17	/CARDIOWR
AB[16:13]	V _{DD}
AB[12:0]	A[12:0]
DB[15:8]	D[23:16]
DB[7:0]	D[31:24]
WE1#	/CARDxCSH
M/R#	V _{DD}
CS#	V _{DD}
BUSCLK	DCLKOUT
BS#	V _{DD}
RD/WR#	/CARDxCSL
RD#	/RD
WE0#	/WE
WAIT#	/CARDxWAIT
RESET#	RESET#

PR31500/PR31700 Host Bus Interface Signals

When the S1D13505 is configured to operate with the PR31500/PR31700, the host interface requires the following signals:

- BUSCLK is a clock input required by the S1D13505 host bus interface. It is separate from the input clock (CLKI) and should be driven by the PR31500/PR31700 bus clock output DCLKOUT.
- Address input AB20 corresponds to the PR31500/PR31700 signal ALE (address latch enable) whose falling edge indicates that the most significant bits of the address are present on the multiplexed address bus (AB[12:0]).
- Address input AB19 should be connected to the PR31500/PR31700 signal /CARDREG. This signal is active when either IO or configuration space of the PR31500/PR31700 PC Card slot is being accessed.
- Address input AB18 should be connected to the PR31500/PR31700 signal /CARDIORD. Either AB18 or the RD# input must be asserted for a read operation to take place.

- Address input AB17 should be connected to the PR31500/PR31700 signal /CARDIOWR. Either AB17 or the WE0# input must be asserted for a write operation to take place.
- Address inputs AB[16:13] and control inputs M/R#, CS# and BS# must be tied to VDD as they are not used in this interface mode.
- Address inputs AB[12:0], and the data bus DB[15:0], connect directly to the PR31500/PR31700 address and data bus, respectively. MD4 must be set to select the proper endian mode on reset (see “S1D13505 Configuration” on page 14). **Because of the PR31500/PR31700 data bus naming convention and endian mode, S1D13505 DB[15:8] must be connected to PR31500/PR31700 D[23:16], and S1D13505 DB[7:0] must be connected to PR31500/PR31700 D[31:24].**
- Control inputs WE1# and RD/WR# should be connected to the PR31500/PR31700 signals /CARDxCSH and /CARDxCSL respectively for byte steering.
- Input RD# should be connected to the PR31500/PR31700 signal /RD. Either RD# or the AB18 input (/CARDIORD) must be asserted for a read operation to take place.
- Input WE0# should be connected to the PR31500/PR31700 signal /WR. Either WE0# or the AB17 input (/CARDIOWR) must be asserted for a write operation to take place.
- WAIT# is a signal output from the S1D13505 that indicates the host CPU must wait until data is ready (read cycle) or accepted (write cycle) on the host bus. Since the host CPU accesses to the S1D13505 may occur asynchronously to the display update, it is possible that contention may occur in accessing the S1D13505 internal registers and/or display buffer. The WAIT# line resolves these contentions by forcing the host to wait until the resource arbitration is complete.

4.4 Direct Connection to the Philips PR31500/PR31700

The S1D13505 was specifically designed to support the Philips MIPS PR31500/PR31700 processor. When configured, the S1D13505 will utilize one of the PC Card slots supported by the processor.

Hardware Description

In this example implementation, the S1D13505 occupies one PC Card slot and resides in the Attribute and IO address range. The processor provides address bits A[12:0], with A[23:13] being multiplexed and available on the falling edge of ALE. Peripherals requiring more than 8K bytes of address space would require an external latch for these multiplexed bits. However, the S1D13505 has an internal latch specifically designed for this processor making additional logic unnecessary. To further reduce the need for external components, the S1D13505 has an optional BUSCLK divide-by-2 feature, allowing the high speed DCLKOUT from the processor to be directly connected to the BUSCLK input of the S1D13505. An optional external oscillator may be used for BUSCLK since the S1D13505 will accept host bus control signals asynchronously with respect to BUSCLK.

The following diagram shows a typical implementation of the interface.

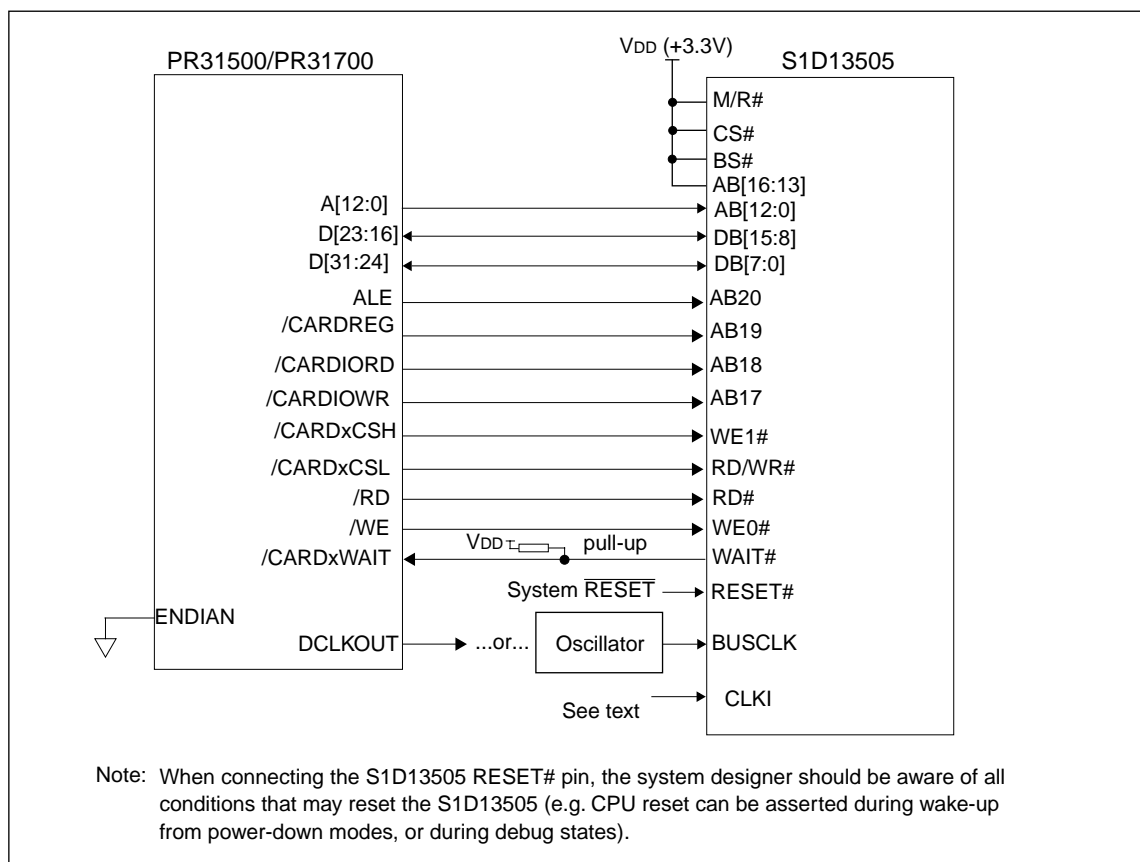


Figure 4-1 Typical Implementation of Direct Connection

The host interface control signals of the S1D13505 are asynchronous with respect to the S1D13505 bus clock. This gives the system designer full flexibility to choose the appropriate source (or sources) for CLKI and BUSCLK. The choice of whether both clocks should be the same, whether to use DCLKOUT as clock source, and whether an external or internal clock divider is needed, should be based on the desired:

- pixel and frame rates.
- power budget.
- part count.
- maximum S1D13505 clock frequencies.

The S1D13505 also has internal CLKI dividers providing additional flexibility.


S1D13505 Configuration

The S1D13505 latches MD15 through MD0 to allow selection of the bus mode and other configuration data on the rising edge of RESET#. For details on configuration, refer to the “*S1D13505 Hardware Functional Specification*”.

The table below shows those configuration settings relevant to the Philips PR31500/PR31700 host bus interface.

Table 4-2 S1D13505 Configuration for Direct Connection

S1D13505 Pin Name	Value on this pin at rising edge of RESET# is used to configure:	
	1 (VDD)	0 (VSS)
MD0	8-bit host bus interface	16-bit host bus interface
MD[3:1]	111 = Philips PR31500/PR31700 host bus interface if Alternate host bus interface is selected	
MD4	Little Endian	Big Endian
MD5	WAIT# is active high (1 = insert wait state)	WAIT# is active low (0 = insert wait state)
MD11	Alternate host bus interface selected	Primary host bus interface selected
MD12	BUSCLK input divided by two: use with DCLKOUT	BUSCLK input not divided: use with external oscillator

 = configuration for Philips PR31500/PR31700 host bus interface

Memory Mapping and Aliasing

The PR31500/PR31700 uses a portion of the PC Card Attribute and IO space to access the S1D13505. The S1D13505 responds to both PC Card Attribute and IO bus accesses, thus freeing the programmer from having to set the PR31500/PR31700 Memory Configuration Register 3 bit CARD1IOEN (or CARD2IOEN if slot 2 is used). As a result, the PR31500/PR31700 sees the S1D13505 on its PC Card slot as described in the table below.

Table 4-3 PR31500/PR31700 to PC Card Slots Address Remapping for Direct Connection

S1D13505 Uses PC Card Slot #	Philips Address	Size	Function
1	0800 0000h	16M byte	Card 1 IO or Attribute
	0900 0000h	8M byte	S1D13505 registers, aliased 4 times at 2M byte intervals
	0980 0000h	8M byte	S1D13505 display buffer, aliased 4 times at 2M byte intervals
	0A00 0000h	32M byte	Card 1 IO or Attribute
	6400 0000h	64M byte	Card 1 Memory
2	0C00 0000h	16M byte	Card 2 IO or Attribute
	0D00 0000h	8M byte	S1D13505 registers, aliased 4 times at 2M byte intervals
	0D80 0000h	8M byte	S1D13505 display buffer, aliased 4 times at 2M byte intervals
	0E00 0000h	32M byte	Card 2 IO or Attribute
	6800 0000h	64M byte	Card 2 Memory

4.5 System Design Using the IT8368E PC Card Buffer

In a system design using one or two ITE IT8368E PC Card and multiple-function IO buffers, the S1D13505 can be interfaced so as to share one of the PC Card slots.

Hardware Description

The IT8368E can be programmed to allocate the same portion of the PC Card Attribute and IO space to the S1D13505 as in the direct connection implementation described in Section 4.4, “*Direct Connection to the Philips PR31500/PR31700*” on page 12.

Following is a block diagram showing an implementation using the IT8368E PC Card buffer.

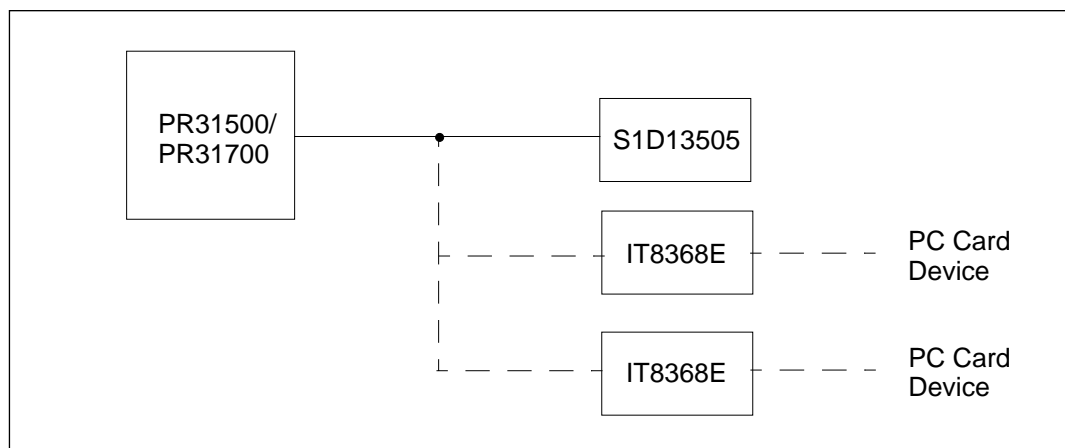


Figure 4-2 IT8368E Implementation Block Diagram

IT8368E Configuration

The ITE IT8368E has been specifically designed to support EPSON LCD/CRT controllers. Older EPSON Controllers not supporting a direct interface to the Philips processor can utilize the IT8368E MFIO pins to provide the necessary control signals, however when using the S1D13505 this is not necessary as the Direct Connection described in Section 4.4, “*Direct Connection to the Philips PR31500/PR31700*” on page 12 can be used.

The IT8368E must have both “Fix Attribute/IO” and “VGA” modes enabled. When both these modes are enabled a 16M byte portion of the system PC Card attribute and IO space is allocated to address the S1D13505.

When the IT8368E senses that the S1D13505 is being accessed, it does not propagate the PC Card signals to its PC Card device. This makes S1D13505 accesses transparent to any PC Card device connected to the same slot.

For mapping details, refer to “*Memory Mapping and Aliasing*” on page 14. For further information on configuring the IT8368E, refer to the “*IT8368E PC Card/GPIO Buffer Chip Specification*”.

S1D13505 Configuration

For details on S1D13505 configuration, see “*S1D13505 Configuration*” on page 14.

4.6 *Software*

Test utilities and Windows® CE v2.0 display drivers are available for the S1D13505. Full source code is available for both the test utilities and the drivers.

The test utilities are configurable for different panel types using a program called 13505CFG, or by directly modifying the source. The Windows® CE v2.0 display drivers can be customized by the OEM for different panel types, resolutions and color depths only by modifying the source.

5 INTERFACING TO THE TOSHIBA MIPS TX3912 PROCESSOR

5.1 Introduction

This application note describes the hardware and software environment necessary to provide an interface between the S1D13505 Embedded RAMDAC LCD/CRT Controller and the Toshiba MIPS TX3912 Processor.

5.2 Interfacing to the TX3912

The Toshiba MIPS TX3912 processor supports up to two PC Card (PCMCIA) slots. It is through this host bus interface that the S1D13505 connects to the TX3912 processor.

The S1D13505 can be successfully interfaced using one of the following configurations:

- Direct connection to the TX3912
- System design using the ITE IT8368E PC Card/GPIO buffer chip

5.3 S1D13505 Host Bus Interface

The S1D13505 implements a 16-bit host bus interface specifically for interfacing to the TX3912 microprocessor.

The TX3912 host bus interface is selected by the S1D13505 on the rising edge of RESET#. After releasing reset, the bus interface signals assume their selected configuration. For S1D13505 configuration, refer to “S1D13505 Configuration” on page 14.

Note: At reset, the Host Interface Disable bit in the Miscellaneous Disable Register (REG[1Bh] bit 7) is set to 1. This means that only REG[1Ah] (read-only) and REG[1Bh] are accessible **until a write to REG[1Bh] sets bit 7 to 0 making all registers accessible**. When debugging a new hardware design, this can sometimes give the appearance that the interface is not working, so it is important to remember to clear this bit before proceeding with debugging.

TX3912 Host Bus Interface Pin Mapping

The following table shows the function of each host bus interface signal.

Table 5-1 TX3912 Host Bus Interface Pin Mapping

S1D13505 Pin Name	Toshiba TX3912
AB20	ALE
AB19	CARDREG*
AB18	CARDIORD*
AB17	CARDIOWR*
AB[16:13]	V _{DD}
AB[12:0]	A[12:0]
DB[15:8]	D[23:16]
DB[7:0]	D[31:24]
WE1#	CARDxCSH*
M/R#	V _{DD}
CS#	V _{DD}
BUSCLK	DCLKOUT
BS#	V _{DD}
RD/WR#	CARDxCSL*
RD#	RD*
WE0#	WE*
WAIT#	CARDxWAIT*
RESET#	PON*

TX3912 Host Bus Interface Signals

When the S1D13505 is configured to operate with the TX3912, the host interface requires the following signals:

- BUSCLK is a clock input required by the S1D13505 host bus interface. It is separate from the input clock (CLKI) and should be driven by the TX3912 bus clock output DCLKOUT.
- Address input AB20 corresponds to the TX3912 signal ALE (address latch enable) whose falling edge indicates that the most significant bits of the address are present on the multiplexed address bus (AB[12:0]).
- Address input AB19 should be connected to the TX3912 signal CARDREG*. This signal is active when either IO or configuration space of the TX3912 PC Card slot is being accessed.
- Address input AB18 should be connected to the TX3912 signal CARDIORD*. Either AB18 or the RD# input must be asserted for a read operation to take place.
- Address input AB17 should be connected to the TX3912 signal CARDIOWR*. Either AB17 or the WE0# input must be asserted for a write operation to take place.
- Address inputs AB[16:13] and control inputs M/R#, CS# and BS# must be tied to VDD as they are not used in this interface mode.
- Address inputs AB[12:0], and the data bus DB[15:0], connect directly to the TX3912 address and data bus, respectively. MD4 must be set to select the proper endian mode on reset (see “S1D13505 Configuration” on page 14). **Because of the TX3912 data bus naming convention and endian mode, S1D13505 DB[15:8] must be connected to TX3912 D[23:16], and S1D13505 DB[7:0] must be connected to TX3912 D[31:24].**
- Control inputs WE1# and RD/WR# should be connected to the TX3912 signals CARDxCSH* and CARDxCSL* respectively for byte steering.
- Input RD# should be connected to the TX3912 signal RD*. Either RD# or the AB18 input (CARDIORD*) must be asserted for a read operation to take place.
- Input WE0# should be connected to the TX3912 signal WR*. Either WE0# or the AB17 input (CARDIOWR*) must be asserted for a write operation to take place.
- WAIT# is a signal output from the S1D13505 that indicates the TX3912 must wait until data is ready (read cycle) or accepted (write cycle) on the host bus. Since the TX3912 accesses to the S1D13505 may occur asynchronously to the display update, it is possible that contention may occur in accessing the S1D13505 internal registers and/or display buffer. The WAIT# line resolves these contentions by forcing the host to wait until the resource arbitration is complete.

5.4 Direct Connection to the Toshiba TX3912

The S1D13505 was specifically designed to support the Toshiba MIPS TX3912 processor. When configured, the S1D13505 will utilize one of the PC Card slots supported by the processor.

Hardware Description

In this example implementation, the S1D13505 occupies one PC Card slot and resides in the Attribute and IO address range. The processor provides address bits A[12:0], with A[23:13] being multiplexed and available on the falling edge of ALE. Peripherals requiring more than 8K bytes of address space would require an external latch for these multiplexed bits. However, the S1D13505 has an internal latch specifically designed for this processor making additional logic unnecessary. To further reduce the need for external components, the S1D13505 has an optional BUSCLK divide-by-2 feature, allowing the high speed DCLKOUT from the processor to be directly connected to the BUSCLK input of the S1D13505. An optional external oscillator may be used for BUSCLK since the S1D13505 will accept host bus control signals asynchronously with respect to BUSCLK.

The following diagram shows a typical implementation of the interface.

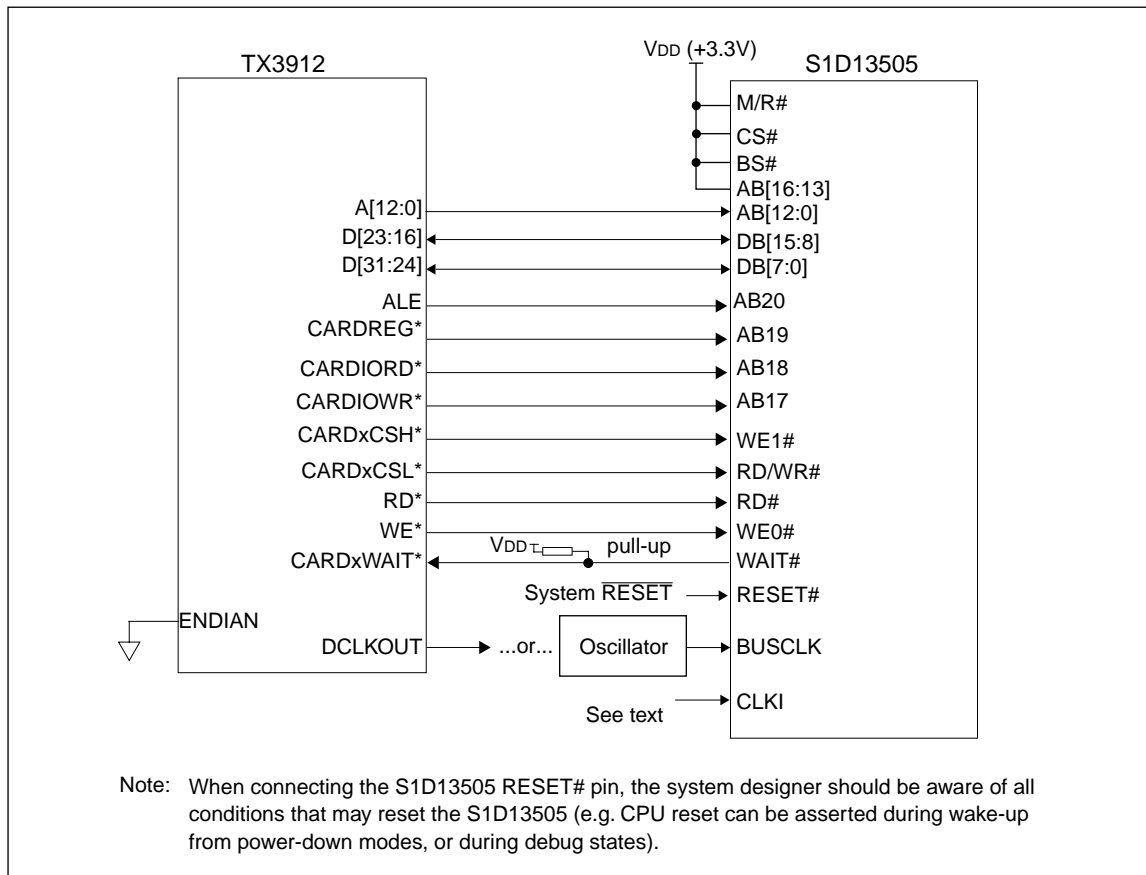


Figure 5-1 Typical Implementation of Direct Connection

The host interface control signals of the S1D13505 are asynchronous with respect to the S1D13505 bus clock. This gives the system designer full flexibility to choose the appropriate source (or sources) for CLKI and BUSCLK. The choice of whether both clocks should be the same, whether to use DCLKOUT as clock source, and whether an external or internal clock divider is needed, should be based on the desired:

- pixel and frame rates.
- power budget.
- part count.
- maximum S1D13505 clock frequencies.

The S1D13505 also has internal CLKI dividers providing additional flexibility.

S1D13505 Configuration

The S1D13505 latches MD15 through MD0 to allow selection of the bus mode and other configuration data on the rising edge of RESET#. For details on configuration, refer to the “*S1D13505 Hardware Functional Specification*”.

The table below shows those configuration settings relevant to the Toshiba TX3912 host bus interface.

Table 5-2 S1D13505 Configuration for Direct Connection

S1D13505 Pin Name	Value on this pin at rising edge of RESET# is used to configure:	
	1 (V _{DD})	0 (V _{SS})
MD0	8-bit host bus interface	16-bit host bus interface
MD[3:1]	111 = Toshiba TX3912 host bus interface if Alternate host bus interface is selected	
MD4	Little Endian	Big Endian
MD5	WAIT# is active high (1 = insert wait state)	WAIT# is active low (0 = insert wait state)
MD11	Alternate host bus interface selected	Primary host bus interface selected
MD12	BUSCLK input divided by two: use with DCLKOUT	BUSCLK input not divided: use with external oscillator

□ = configuration for Toshiba TX3912 host bus interface

Memory Mapping and Aliasing

The TX3912 uses a portion of the PC Card Attribute and IO space to access the S1D13505. The S1D13505 responds to both PC Card Attribute and IO bus accesses, thus freeing the programmer from having to set the TX3912 Memory Configuration Register 3 bit CARD1IOEN (or CARD2IOEN if slot 2 is used). As a result, the TX3912 sees the S1D13505 on its PC Card slot as described in the table below.

Table 5-3 TX3912 to PC Card Slots Address Remapping for Direct Connection

S1D13505 Uses PC Card Slot #	Toshiba Address	Size	Function
1	0800 0000h	16M byte	Card 1 IO or Attribute
	0900 0000h	8M byte	S1D13505 registers, aliased 4 times at 2M byte intervals
	0980 0000h	8M byte	S1D13505 display buffer, aliased 4 times at 2M byte intervals
	0A00 0000h	32M byte	Card 1 IO or Attribute
	6400 0000h	64M byte	Card 1 Memory
2	0C00 0000h	16M byte	Card 2 IO or Attribute
	0D00 0000h	8M byte	S1D13505 registers, aliased 4 times at 2M byte intervals
	0D80 0000h	8M byte	S1D13505 display buffer, aliased 4 times at 2M byte intervals
	0E00 0000h	32M byte	Card 2 IO or Attribute
	6800 0000h	64M byte	Card 2 Memory

5.5 System Design Using the IT8368E PC Card Buffer

In a system design using one or two ITE IT8368E PC Card and multiple-function IO buffers, the S1D13505 can be interfaced so as to share one of the PC Card slots.

Hardware Description

The IT8368E can be programmed to allocate the same portion of the PC Card Attribute and IO space to the S1D13505 as in the direct connection implementation described in Section 4.4, “*Direct Connection to the Philips PR31500/PR31700*” on page 12.

Following is a block diagram showing an implementation using the IT8368E PC Card buffer.

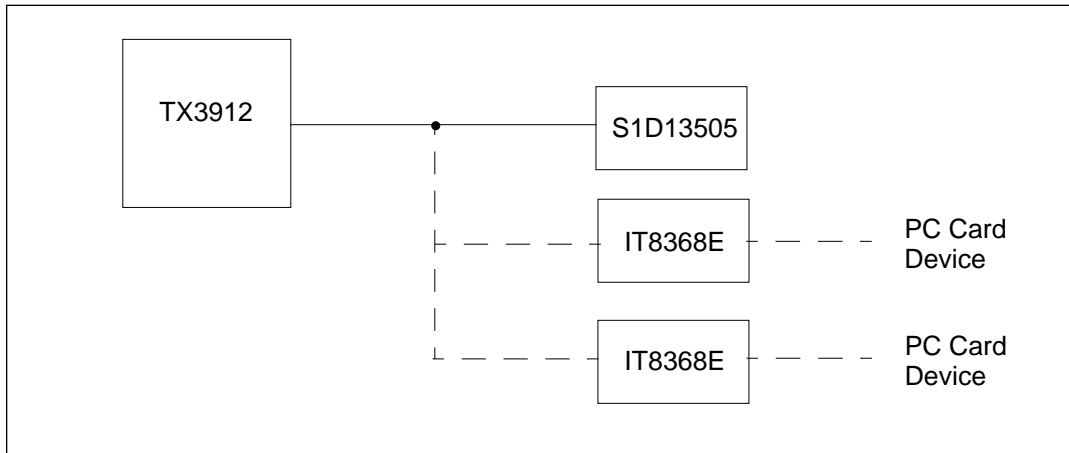


Figure 5-2 IT8368E Implementation Block Diagram

IT8368E Configuration

The ITE IT8368E has been specifically designed to support EPSON LCD/CRT controllers. Older EPSON Controllers not supporting a direct interface to the Toshiba processor can utilize the IT8368E MFIO pins to provide the necessary control signals, however when using the S1D13505 this is not necessary as the Direct Connection described in Section 4.4, “*Direct Connection to the Philips PR31500/PR31700*” on page 12 can be used.

The IT8368E must have both “Fix Attribute/IO” and “VGA” modes enabled. When both these modes are enabled a 16M byte portion of the system PC Card attribute and IO space is allocated to address the S1D13505.

When the IT8368E senses that the S1D13505 is being accessed, it does not propagate the PC Card signals to its PC Card device. This makes S1D13505 accesses transparent to any PC Card device connected to the same slot.

For mapping details, refer to “*Memory Mapping and Aliasing*” on page 14. For further information on configuring the IT8368E, refer to the “*IT8368E PC Card/GPIO Buffer Chip Specification*”.

S1D13505 Configuration

For S1D13505 configuration, refer to “*S1D13505 Configuration*” on page 14.

5.6 *Software*

Test utilities and Windows® CE v2.0 display drivers are available for the S1D13505. Full source code is available for both the test utilities and the drivers.

The test utilities are configurable for different panel types using a program called 13505CFG, or by directly modifying the source. The Windows® CE v2.0 display drivers can be customized by the OEM for different panel types, resolutions and color depths only by modifying the source.

6 INTERFACING TO THE MOTOROLA MPC821 MICROPROCESSOR

6.1 Introduction

This application note describes the hardware and software necessary to provide an interface between the S1D13505 Embedded RAMDAC LCD/CRT Controller and the Motorola MPC821 Processor.

For further information on the “*S1D13505 refer to its Hardware Functional Specification*”. For information on the Motorola MPC821 Processor contact the Motorola Design Line or your local Motorola sales office.

General Description

The S1D13505 provides native Power PC bus support making it very simple to interface the two devices. This application note describes, both the environment necessary to connect the S1D13505 to the MPC821 native system bus, and the connection between the S5U13505P00C Evaluation Board and the Motorola MPC821 Application Development System (ADS).

Additionally, by implementing a dedicated display buffer, the S1D13505 can reduce system power consumption, improve image quality, and increase system performance as compared to the MPC821's on-chip LCD controller.

6.2 Hardware Connections

The S1D13505 implements a native MPC8xx bus interface mode and through the use of the MPC821 chip selects, can share the system bus with all other MPC821 peripherals. Figure 6-1 demonstrates a typical implementation of the interface.

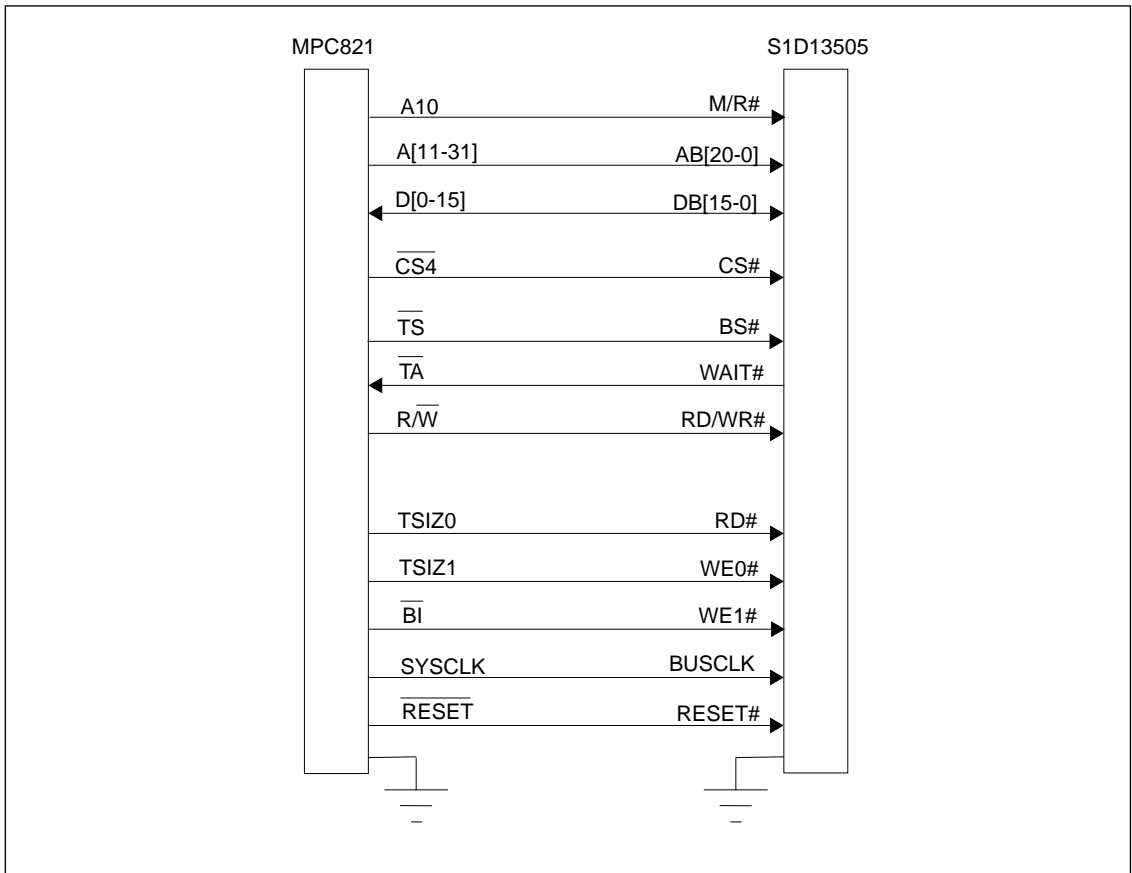


Figure 6-1 Schematic for MPC821/S1D13505 Interface

Table 6-1 shows the connections between the pins and signals of the MPC821 and the S1D13505.

Note: The interface was designed using a Motorola MPC821 Application Development System (ADS). The ADS board has 5 volt logic connected to the data bus, so the interface included two 74F245 octal buffers on the D[0:15] between the ADS and the S1D13505. In a true 3 volt system, no buffering is necessary.

Table 6-1 List of Connections from MPC821ADS to S1D13505

MPC821 Signal Name#1	MPC821ADS Connector and Pin Name	S1D13505 Signal Name
Vcc	P6-A1, P6-B1	Vcc
A10	P6-C23	M/R#
A11	P6-A22	AB20
A12	P6-B22	AB19
A13	P6-C21	AB18
A14	P6-C20	AB17
A15	P6-D20	AB16
A16	P6-B24	AB15
A17	P6-C24	AB14
A18	P6-D23	AB13
A19	P6-D22	AB12
A20	P6-D19	AB11
A21	P6-A19	AB10
A22	P6-D28	AB9
A23	P6-A28	AB8
A24	P6-C27	AB7
A25	P6-A26	AB6
A26	P6-C26	AB5
A27	P6-A25	AB4
A28	P6-D26	AB3
A29	P6-B25	AB2
A30	P6-B19	AB1
A31	P6-D17	AB0
D0	P12-A9	DB15
D1	P12-C9	DB14
D2	P12-D9	DB13
D3	P12-A8	DB12
D4	P12-B8	DB11
D5	P12-D8	DB10
D6	P12-B7	DB9
D7	P12-C7	DB8
D8	P12-A15	DB7
D9	P12-C15	DB6
D10	P12-D15	DB5
D11	P12-A14	DB4
D12	P12-B14	DB3
D13	P12-D14	DB2
D14	P12-B13	DB1
D15	P12-C13	DB0
SRESET	P9-D15	RESET#
SYSCLK	P9-C2	BUSCLK
CS4	P6-D13	CS#
SRESET	P9-D15	RESET#
SYSCLK	P9-C2	BUSCLK
CS4	P6-D13	CS#
TS	P6-B7	BS#
TA	P6-B6	WAIT#
R/W	P6-D8	RD/WR#
TSIZ0	P6-B18	RD#
TSIZ1	P6-C18	WE0#
BI	P6-B9	WE1#
Gnd	P12-A1, P12-B1, P12-A2, P12-B2, P12-A3, P12-B3, P12-A4, P12-B4, P12-A5, P12-B5, P12-A6, P12-B6, P12-A7	Vss

#1: Note that the bit numbering of the PowerPC bus signals is reversed. e.g. the most significant address bit is A0, the next is A1, A2, etc.

6.3 Hardware Configuration

S1D13505 Configuration

The S1D13505 uses MD0 through MD15 to allow selection of the bus mode and other configuration data on the rising edge of RESET#. Table 6-2 shows the settings used for the S1D13505 in this interface.

Table 6-2 S1D13505 Configuration Settings

Signal	1	0
MD0	8-bit host bus interface	16-bit host bus interface
MD1	See "Host Bus Selection" table below	See "Host Bus Selection" table below
MD2		
MD3		
MD4	Little Endian	Big Endian
MD5	Wait# signal is active high	Wait# signal is active low
MD6	See "Memory Configuration" table below	See "Memory Configuration" table below
MD7		
MD8		
MD9	Reserved	Configure SUSPEND# pin as Hardware Suspend Enable
MD10	Active low (On) LCDPWR / GPO polarity	Active high (On) LCDPWR / GPO polarity
MD11	Alternate host bus interface	Primary host bus interface
MD12	Reserved	Reserved
MD13	Reserved	Reserved
MD14	Reserved	Reserved
MD15	Reserved	Reserved

= required settings for MPC821 support.

Table 6-3 Host Bus Selection

MD11	MD3	MD2	MD1	Option	Host Bus Interface
0	0	0	0	1	SH-3/SH-4 bus interface
0	0	0	1	2	MC68K Bus 1
0	0	1	0	3	MC68K Bus 2
0	0	1	1	4	Generic
0	1	0	0	5	Reserved
0	1	0	1	6	MIPS/ISA
0	1	1	0	7	Power PC
0	1	1	1	8	PC Card (PCMCIA)
1	x	x	x	9+	Reserved

Table 6-4 Memory Configuration

MD7	MD6	Option	Memory Selection
0	0	1	Symmetrical 256K x 16 DRAM
0	1	2	Symmetrical 1M x 16 DRAM
1	0	3	Asymmetrical 256K x 16 DRAM
1	1	4	Asymmetrical 1M x 16 DRAM

MPC821 Chip Select Configuration

The DRAM on the MPC821 ADS board extends from address 0 through 0x3ffff, so the S1D13505 was addressed starting at 0x400000. A total of 4M bytes of address space is used, where the lower 2M bytes is reserved for the S1D13505 on-chip registers and the upper 2M bytes is used to access the S1D13505 display buffer.

Chip select 4 was used to control the S1D13505. The following options were selected in the base address register (BR4):

- BA (0:16) = 0000 0000 0100 0000 0
 - set starting address of S1D13505 to 0x40 0000
- AT (0:2) = 0
 - ignore address type bits
- PS (0:1) = 1:0
 - memory port size is 16 bits
- PARE = 0
 - disable parity checking
- WP = 0
 - disable write protect
- MS (0:1) = 0:0
 - select General Purpose Chip Select module to control this chip select
- V = 1
 - set valid bit to enable chip select

The following options were selected in the option register (OR4):

- AM (0:16) = 1111 1111 1100 0000 0
 - mask all but upper 10 address bits; S1D13505 consumes 4M byte of address space
- ATM (0:2) = 0
 - ignore address type bits
- CSNT = 0
 - normal $\overline{\text{CS}}/\overline{\text{WE}}$ negation
- ACS (0:1) = 1:1
 - delay $\overline{\text{CS}}$ assertion by \int clock cycle from address lines
- BI = 0
 - do not assert Burst Inhibit
- SCY (0:3) = 0
 - wait state selection; this field is ignored since external transfer acknowledge is used; see SETA below
- SETA = 1
 - the S1D13505 generates an external transfer acknowledge using the WAIT# line
- TRLX = 0
 - normal timing
- EHTR = 0
 - normal timing

6.4 Test Software

The test software is very simple. It configures chip select 4 (CS4) on the MPC821 to map the S1D13505 to an unused 4M byte block of address space. Next, it loads the appropriate values into the option register for CS4 and writes the value 0 to the S1D13505 register REG[1Bh] to enable the S1D13505 host interface. Lastly, the software runs a tight loop that reads the S1D13505 Revision Code Register REG[00h]. This allows monitoring of the bus timing on a logic analyzer.

Test Software Source Code

The following source code was entered into the memory of the MPC821ADS using the line-by-line assembler in MPC8BUG, the debugger provided with the ADS board.¹ Once the program was executed on the ADS, a logic analyzer was used to verify operation of the interface hardware.

It is important to note that when the MPC821 comes out of reset, it's on-chip caches and MMU are disabled. If the data cache is enabled, then the MMU must be set up so that the S1D13505 memory block is tagged as non-cacheable. This ensures the MPC821 does not attempt to cache any data read from, or written to, the S1D13505 or its display refresh buffer.

```

BR4      equ      $120          ; CS4 base register
OR4      equ      $124          ; CS4 option register
MemStart equ      $40           ; upper word of S1D13505 start address
DisableReg equ     $1b          ; address of S1D13505 Disable Register
RevCodeReg equ     0            ; address of Revision Code Register

Start    mfspr     r1,IMMR       ; get base address of internal registers
        andis.    r1,r1,$ffff    ; clear lower 16 bits to 0
        andis.    r2,r0,0        ; clear r2
        oris      r2,r2,MemStart ; write base address
        ori       r2,r2,$0801    ; port size 16 bits; select GPCM; enable
        stw       r2,BR4(r1)    ; write value to base register
        andis.    r2,r0,0        ; clear r2
        oris      r2,r2,$ffc0    ; address mask – use upper 10 bits
        ori       r2,r2,$0608    ; normal CS negation; delay CS ½ clock;
                                ; no burst inhibit (13505 does this)
        stw       r2,OR4(r1)    ; write to option register
        andis.    r1,r0,0        ; clear r1
        oris      r1,r1,MemStart ; point r1 to start of S1D13505 mem space
        stb       r1,DisableReg(r1) ; write 0 to disable register
Loop     lbz       r0,RevCodeReg(r1) ; read revision code into r1
        b         Loop          ; branch forever

        end

```

1. MPC8BUG does not support comments or symbolic equates; these have been added for clarity.

7 DAC APPLICATION NOTES

7.1 DAC Application Notes

Introduction

A video-DAC for CRT display is built-in the S1D13505. It operates with something like analog data, which is very different from other logic parts operated digitally. When operating the DAC normally, some checks are needed. Furthermore, even though the DAC does not need to operate when displaying on a LCD panel and not on a CRT display, some checks are needed when operating the S1D13505. This document describes DAC.

Please check these notes for normal S1D13505 operation.

Notes When Operating the S1D13505 Built-in DAC

Please check these notes when operating the S1D13505 built-in DAC, as described in the following sections:

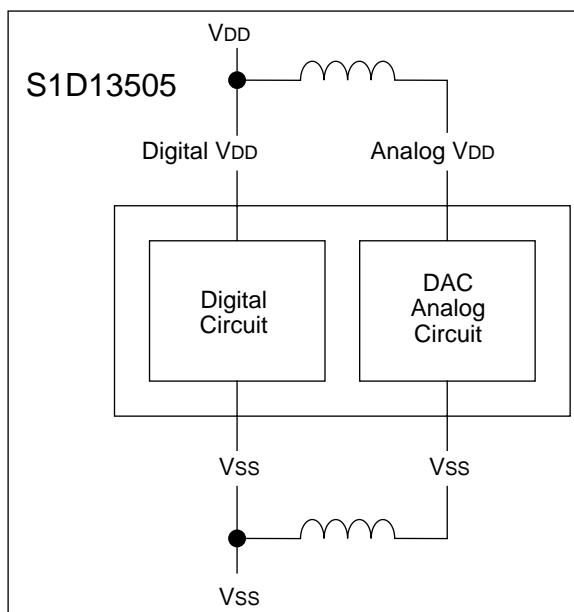
DAC Isolated Power Source

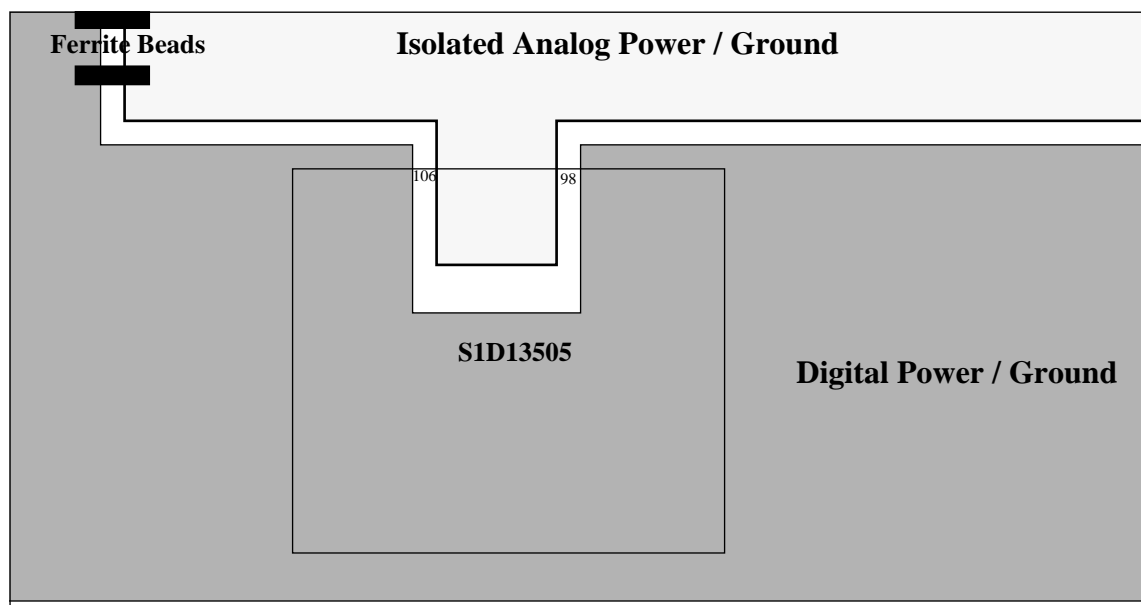
Table 7-1 shows power source required to operate the S1D13505.

Table 7-1 S1D13505 Power Supply Pin Description

Power Source Pin Name	Power Source Specification
VDD	Uses for digital system power source of internal logic and I/O cells.
VSS	Uses for digital system ground of internal logic and I/O cells.
DACVDD	Built-in DAC isolated power source
DAVVSS	Built-in DAC isolated ground

The power source and ground of digital and analog systems are separated in the S1D13505, because if they are used commonly at the S1D13505, a digital power source system noise occurs and the analog characteristics of the DAC can not be maintained. Therefore, the external part of the S1D13505 on the board between digital and analog systems must be separated. To achieve the separation, we recommend that digital and analog systems be physically enclosed by wiring on the board or noise should be stopped by using ferrite beads, a choke coil or a noise filter.





All analog parts should be located in the area of the analog power system, separate from the digital power system.

Peripheral Circuit of DAC Pins

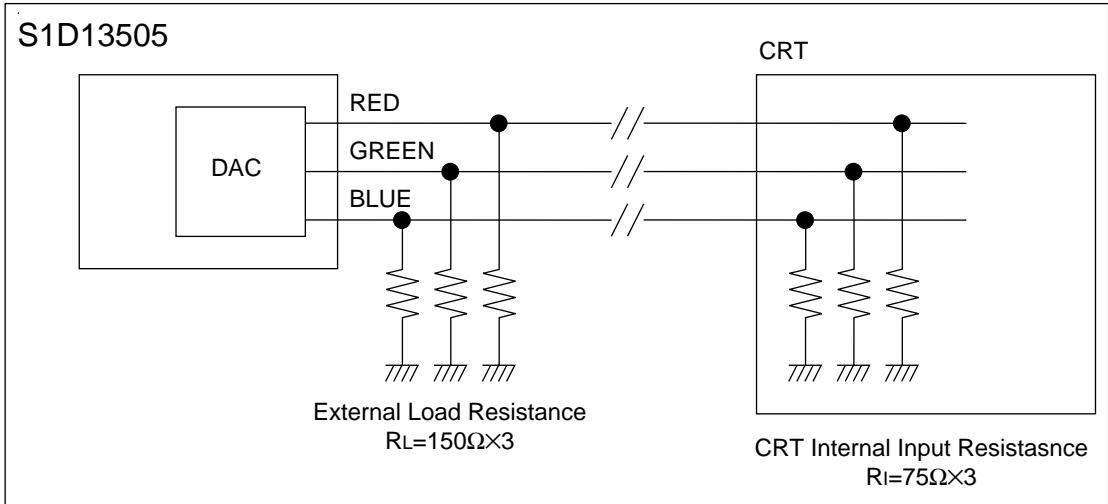
There are pins for the DAC in the S1D13505, as shown in Table 7-2. Each pin should be used where it is necessary to construct a circuit.

Table 7-2 S1D13505 DAC Pin Description

Pin Name	Functions	Necessary Circuit
RED	Analog red (red) signal output pin	Load resistance
GREEN	Analog green (green) signal output pin	Load resistance
BLUE	Analog blue (blue) signal output pin	Load resistance
IREF	Standard current source connecting pin	Constant-current source circuit
HRTC	Retrace signal output pin in the horizontal direction	—
VRTC	Retrace signal output pin in the vertical direction	—

RED/GREEN/BLUE Pins

These pins are used for RGB analog signals to output a CRT display. As shown in the following figures, when the combined resistance of $50\ \Omega$ is the total load of the external resistance of $150\ \Omega$ and the internal resistance of $75\ \Omega$ of the CRT display connected to the S1D13505, the peak level of $0.7\ \text{V}$ is outputted. Therefore, the load resistance of $150\ \Omega$ should be connected to the RGB output pins of DAC in the S1D13505 and the output level is adjusted correctly.

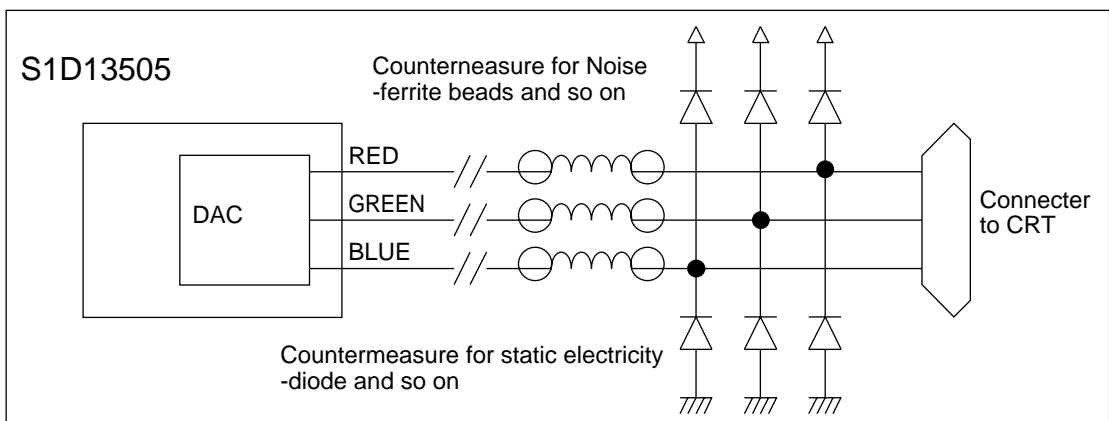


Do not insert a part to change the resistance to maintain the analog characteristics of these RGB analog signal outputs. Except for the application device's built-in CRT display, a general type of CRT display may be used as an external device connecting to another device. In this case, the RGB analog pins of the DAC become external pins and the outputs of the DAC interface directly to the external device. Therefore, when composing an application device, the RGB pins may require some countermeasures to prevent EMI and static electricity noise.

The RGB output pins should be enclosed to prevent noise in an application device or an external device. However, do not use a part that changes its resistance. When the RGB output pins need to be enclosed, use ferrite beads, a choke coil or a noise filter.

The DAC of the S1D13505 can be protected from general levels of static electricity. However, the protection level of the DAC is set in an IC (Integrated Circuit) and it is not set in the device level. Therefore, when the RGB pins of the DAC are external pins, special countermeasures to static electricity are necessary to prevent malfunctions or breakdowns caused by static electricity.

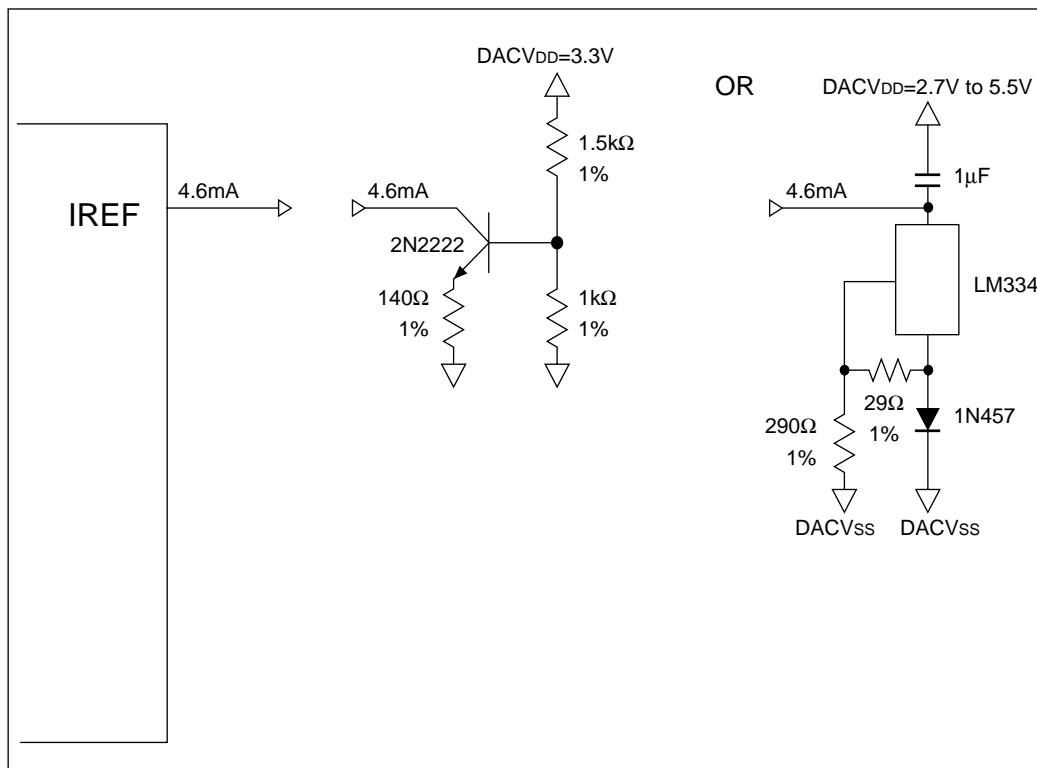
One effective countermeasures to insert a diode for the static electricity into each RGB pin.



IREF Pin

As the IREF pin is used to adjust the standard operation current of the DAC, the normal constant-current should flow from the IREF pin to ground. To output 0.7 V from the RGB pins at the peak level, 4.6mA of normal constant-current is needed. Therefore, the constant-current circuit through which the current flows to ground must be connected to the IREF pin.

The following circuit diagram shows an example of a 4.6 mA constant current circuit. The most important thing is the 4.6 mA constant-current circuit. If other parts and their constant values need to be changed, they can be composed at the circuit.



The current value of the IREF pin is in proportion to the RGB analog output level. Therefore, when the current is more than 4.6 mA, the output level of 0.7 V goes up and the luminance of the CRT display also goes up. This means that the luminance displayed on the CRT can be adjusted by the current at the IREF pin. If the constant-current value is changed from 4.6 mA to adjust the luminance, take care with the input level of the RGB pin of the CRT display.

HRTC and VRTC Pins

The FRTC and VRTC signals do not need a peripheral circuit. Connect them directly to the input signal of the CRT.

However, if the CRT display is used for the external device, the countermeasure may be needed to noise of EMI or static electricity, because both the FRCT and VRTC signals are used directly for the external pins in the same way as the RGB pins. As non-analog characteristics of both pins-FRTC and VRTC are the same as the RGB pins, a resistance should be inserted to the pins according to need for the countermeasure.

Notes When the DAC is Not Used

Even when the DAC is not used, the S1D13505 may malfunction if the power system and pins of the DAC are correctly set.

When the DAC is not used, note the points described in the following sections.

Isolated DAC Power Pin

The DACVDD and DACVSS of the analog power system must be connected to the VDD and VSS, respectively, of the digital power system. Never set the power system to be floating even when the DAC is not used, because the internal circuit of the S1D13505 is not stable and it may malfunction.

When the DAC is connected to the power of the digital system, the noise-proof facility described in “DAC Isolated Power Source”, -for example, the DAC enclosed by wiring on the board or using the noise filter, is not needed. Even if the DAC is connected directly to the digital power system, there is no problem because it is most important that the electric potential in the internal circuit is stable.

Usually, the DAC built-into the S1D13505 is set to disable automatically by the internal signal, except when using the DAC when the CRT mode is enabled.

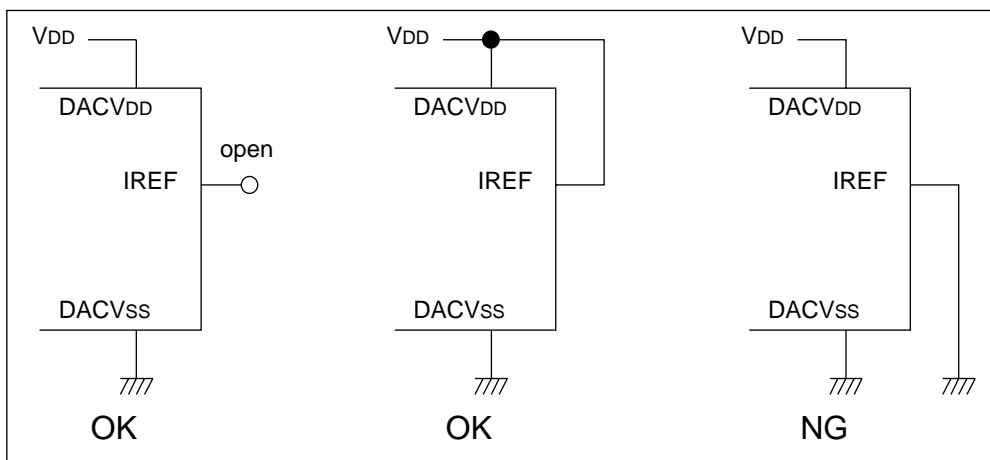
Futhermore, the internal circuit is set to be disabled and static. This is why no current flows even if the power system of the DAC is connected to the power of the digital system when the DAC is not used.

Isolated DAC Signal Pins

When the DAC built-in S1D13505 is not used, the isolated DAC signal pins should be set as shown in the following table.

Pin Name	How to Set
RED	Open, NC
GREEN	Open, NC
BLUE	Open, NC
IREF	Open, NC
HRTC	Open, NC
VRTC	Open, NC

As these analog RGB outputs, HRTC and VRTC signals are used for the isolated output pins, even if they are set open and connected to nothing, no trouble will happen. However, care must be taken when opening the IREF, because the IREF pin is used to conduct the IREF current to ground. When the DAC is not used, if the IREF pin is grounded, more current than the normal capacity flows from the DACVDD through the IREF pin to the DACVss. In this case, the wiring in the internal circuit may be damaged by over current. Furthermore, when the CRT display is disabled, the IREF circuit of the DAC is disabled by the internal signal. Therefore, if the S1D13505 is normally controlled, no over-capacity current must be generated. However, the IREF pin must be open, because if the setting is mistaken, it is very dangerous. Furthermore, although the IREF pin may be connected to the DACVDD, never connect it to the DACVss.



8 POWER CONSUMPTION

8.1 S1D13505 Power Consumption

S1D13505 power consumption is affected by many system design variables.

- Input clock frequency (CLKI):
The CLKI frequency determines the LCD frame-rate, CPU performance to memory, and other functions – the higher the input clock frequency, the higher the frame-rate, performance and power consumption.
- CPU interface:
The S1D13505 current consumption depends on the BUSCLK frequency, data width, number of toggling pins, and other factors – the higher the BUSCLK, the higher the CPU performance and power consumption.
- VDD voltage level:
The voltage level affects power consumption – the higher the voltage, the higher the consumption.
- Display mode:
The resolution and color depth affect power consumption – the higher the resolution/color depth, the higher the consumption.
- Internal CLK divide:
Internal registers allow the input clock to be divided before going to the internal logic blocks – the higher the divide, the lower the power consumption.

There are two power save modes in the S1D13505: Software and Hardware SUSPEND. The power consumption of these modes is affected by various system design variables.

- DRAM refresh mode (CBR or self-refresh):
Self-refresh capable DRAM allows the S1D13505 to disable the internal memory clock thereby saving power.
- CPU bus state during SUSPEND:
The state of the CPU bus signals during SUSPEND has a substantial effect on power consumption. An inactive bus (e.g. BUSCLK = low, Addr = low etc.) reduces overall system power consumption.
- CLKI state during SUSPEND:
Disabling the CLKI during SUSPEND has substantial power savings.

Conditions

Table 8-1 below gives an example of a specific environment and its effects on power consumption.

Table 8-1 S1D13505 Total Power Consumption

Test Condition $V_{DD} = 3.3V$ ISA Bus (8MHz)		Gray Shades / Colors	Total Power Consumption		
			Active	Power Save Mode	
				Software	Hardware
1	Input Clock = 6MHz LCD Panel = 320x240 4-bit Single Monochrome	Black-and-White 4 Gray Shades 16 Gray Shades	18.6mW 20.3mW 22.8mW	4.29mW*1	0.33μW*2
2	Input Clock = 6MHz LCD Panel = 320x240 8-bit Single Color	4 Colors 16 Colors 256 Colors	22.3mW 25.3mW 29.0mW	4.32mW*1	0.33μW*2
3	Input Clock = 25MHz LCD Panel = 640x480 8-bit Dual Monochrome	Black-and-White 16 Gray Shades	58.5mW 71.7mW	5.71mW*1	0.33μW*2
4	Input Clock = 25MHz LCD Panel = 640x480 16-bit Dual Color	16 Colors 256 Colors 64K Colors	93.4mW 98.1mW 101.3mW	5.74mW*1	0.33μW*2
5	Input Clock = 33.333MHz CRT = 640x480 Color	16 Colors 256 Colors 64K Colors	221.1mW 234.0mW 237.3mW	6.34mW*1	0.33μW*2

Notes: *1. Conditions for Software SUSPEND:

- CPU interface active (signals toggling)
- CLKI active
- Self-Refresh DRAM

*2. Conditions for Hardware SUSPEND:

- CPU interface inactive (high impedance)
- CLKI stopped
- Self-Refresh DRAM

8.2 Summary

The system design variables in Section 8.1, “S1D13505 Power Consumption” and in Table 8-1 show that S1D13505 power consumption depends on the specific implementation. Active Mode power consumption depends on the desired CPU performance and LCD frame-rate, whereas Power Save Mode consumption depends on the CPU Interface and Input Clock state.

In a typical design environment, the S1D13505 can be configured to be an extremely power-efficient LCD Controller with high performance and flexibility.

S1D13505F00A
Embedded RAMDAC LCD/CRT Controller

Windows® CE
Display Drivers

Table of Contents

1 WINDOWS® CE DISPLAY DRIVERS.....6-1

1.1 Program Requirements6-1

1.2 Example Driver Builds6-1

 Build for the Hitachi D9000 and ETMA ODO Evaluation Systems6-1

 Build for CEPC (X86)6-3

1.3 Example Installation6-5

 Installation for Hitachi D9000 and ETMA ODO6-5

 Installation for CEPC Environment6-5

1.4 Comments6-6

1 WINDOWS® CE DISPLAY DRIVERS

The Windows® CE display drivers are designed to support the S1D13505 Embedded RAMDAC LCD/CRT Controller running under the Microsoft Windows® CE operating system. Available drivers include: 4, 8 and 16 bit-per-pixel landscape modes, and 8 and 16 bit-per-pixel portrait modes.

For updated source code, visit Epson R&D on the World Wide Web at www.erd.epson.com, or contact your Seiko Epson or Epson Electronics America sales representative.

1.1 Program Requirements

Video Controller	: S1D13505
Display Type	: LCD or CRT
Windows Version	: CE Version 2.0

1.2 Example Driver Builds

Build for the Hitachi D9000 and ETMA ODO Evaluation Systems

To build a Windows® CE v2.0 display driver for the Hitachi D9000 or ETMA ODO platform, follow the instructions below. The instructions assume the S5U13505P00C-D9000 evaluation board is plugged into slots 6 and 7 on the D9000/ODO platform, and the SEIKO EPSON common interface FPGA (ODO.RBF) is used to interface with the S1D13505.

1. Install Microsoft Windows NT v4.0.
2. Install Microsoft Visual C/C++ v5.0.
3. Install the Microsoft Windows® CE Embedded Toolkit (ETK) by running SETUP.EXE from the ETK compact disc #1.
4. Create a new project by following the procedure documented in “Creating a New Project Directory” from the Windows® CE ETK v2.0. Alternately, use the current “DEMO7” project included with the ETK v2.0. Follow the steps below to create a “SH3 DEMO7” shortcut on the Windows NT v4.0 desktop which uses the current “DEMO7” project:
 - a. Right click on the “Start” menu on the taskbar.
 - b. Click on the item “Open All Users” and the “Start Menu” window will come up.
 - c. Click on the icon “Programs”.
 - d. Click on the icon “Windows® CE Embedded Development Kit”.
 - e. Drag the icon “SH3 DEMO1” onto the desktop using the right mouse button.
 - f. Click on “Copy Here”.
 - g. Rename the icon “SH3 DEMO1” on the desktop to “SH3 DEMO7” by right clicking on the icon and choosing “rename”.
 - h. Right click on the icon “SH3 DEMO7” and click on “Properties” to bring up the “SH3 DEMO7 Properties” window.
 - i. Replace the string “DEMO1” under the entry “Target” with “DEMO7”.
5. Create a sub-directory named S1D13505 under \wince\platform\odo\drivers\display.
6. Copy the source code to the S1D13505 subdirectory.
7. Add an entry for the S1D13505 in the file \wince\platform\odo\drivers\display\dirs.

8. Modify the file PLATFORM.BIB (using any text editor such as NOTEPAD) to set the default display driver to the file S1D13505.DLL. S1D13505.DLL will be created during the build in step 12. Note that PLATFORM.BIB is located in X:\wince\platform\odo\files (where X: is the drive letter).

You may replace the following lines in PLATFORM.BIB:

```
IF ODO_NODISPLAY !
IF ODO_DISPLAY_CITIZEN_8BPP
ddi.dll    $(_FLATRELEASEDIR)\citizen.dll    NK SH
ENDIF
IF ODO_DISPLAY_CITIZEN_2BPP
ddi.dll    $(_FLATRELEASEDIR)\citizen.dll    NK SH
ENDIF
IF ODO_DISPLAY_CITIZEN_8BPP !
IF ODO_DISPLAY_CITIZEN_2BPP !
ddi.dll    $(_FLATRELEASEDIR)\odo2bpp.dll    NK SH
ENDIF
ENDIF
ENDIF
```

with this line:

```
ddi.dll    $(_FLATRELEASEDIR)\SED1355.dll    NK SH
```

9. Edit the file MODE.H (located in X:\wince\platform\odo\drivers\display\SED1355) to set the desired screen resolution, color depth (bpp) and panel type. The sample code defaults to a 640x480 color dual passive 16-bit LCD panel. To support one of the other listed panels, change the #define statement.
10. Edit the file PLATFORM.REG to set the same screen resolution and color depth (bpp) as in MODE.H. PLATFORM.REG is located in X:\wince\platform\odo\files. The display driver section of PLATFORM.REG should be:

```
; Default for EPSON Display Driver
; 640x480 at 8bits/pixel
; Useful Hex Values
; 1024=0x400, 768=0x300 640=0x280 480=0x1E0 320=140 240=0xF0
[HKEY_LOCAL_MACHINE\Drivers\Display\SED1355]
"CxScreen"=dword:280
"CyScreen"=dword:1E0
"Bpp"=dword:8
```

11. Generate the proper building environment by double-clicking on the sample project icon (i.e., SH3 DEMO7).
12. Type BLDDemo <ENTER> at the DOS prompt of the SH3 DEMO7 window to generate a Windows® CE image file (NK.BIN).

Build for CEPC (X86)

To build a Windows® CE v2.0 display driver for the CEPC (X86) platform using a S5U13505P00C evaluation board, follow the instructions below:

1. Install Microsoft Windows NT v4.0.
2. Install Microsoft Visual C/C++ v5.0.
3. Install the Microsoft Windows® CE Embedded Toolkit (ETK) by running SETUP.EXE from the ETK compact disc #1.
4. Create a new project by following the procedure documented in “Creating a New Project Directory” from the Windows® CE ETK v2.0. Alternately, use the current “DEMO7” project included with the ETK v2.0. Follow the steps below to create a “X86 DEMO7” shortcut on the Windows NT v4.0 desktop which uses the current “DEMO7” project:
 - a. Right click on the “Start” menu on the taskbar.
 - b. Click on the item “Open All Users” and the “Start Menu” window will come up.
 - c. Click on the icon “Programs”.
 - d. Click on the icon “Windows® CE Embedded Development Kit”.
 - e. Drag the icon “X86 DEMO1” onto the desktop using the right mouse button.
 - f. Click on “Copy Here”.
 - g. Rename the icon “X86 DEMO1” on the desktop to “X86 DEMO7” by right clicking on the icon and choosing “rename”.
 - h. Right click on the icon “X86 DEMO7” and click on “Properties” to bring up the “X86 DEMO7 Properties” window.
 - i. Replace the string “DEMO1” under the entry “Target” with “DEMO7”.
 - j. Click on “OK” to finish.
5. Create a sub-directory named S1D13505 under \wince\platform\cepc\drivers\display.
6. Copy the source code to the S1D13505 subdirectory.
7. Add an entry for the S1D13505 in the file \wince\platform\cepc\drivers\display\dirs.
8. Modify the file CONFIG.BIB (using any text editor such as NOTEPAD) to set the system RAM size, the S1D13505 IO port and display buffer address mapping. Note that CONFIG.BIB is located in X:\wince\platform\cepc\files (where X: is the drive letter). Since the S5U13505P00C maps the IO port to 0xE00000 and memory to 0xC00000, the CEPC machine should use the CMOS setup to create a 4M byte hole from address 0xC00000 to 0xFFFFF. The following lines should be in CONFIG.BIB:

```
NK 80200000 00500000 RAMIMGE
```

```
RAM 80700000 00500000 RAM
```

Note:

S1D13505.H should include the following:

```
#define PhysicalVmemSize 0x00200000L
```

```
#define PhysicalPortAddr 0x00E00000L
```

```
#define PhysicalVmemAddr 0x00C00000L
```

9. Edit the file PLATFORM.BIB (located in X:\wince\platform\cepc\files) to set the default display driver to the file S1D13505.DLL. S1D13505.DLL will be created during the build in step 13.

You may replace the following lines in PLATFORM.BIB:

```
IF CEPC_DDI_VGA2BPP
ddi.dll    $(_FLATRELEASEDIR)\ddi_vga2.dll    NK SH
ENDIF

IF CEPC_DDI_VGA8BPP
ddi.dll    $(_FLATRELEASEDIR)\ddi_vga8.dll    NK SH
ENDIF

IF CEPC_DDI_VGA2BPP !
IF CEPC_DDI_VGA8BPP !
ddi.dll    $(_FLATRELEASEDIR)\ddi_s364.dll    NK SH
ENDIF
ENDIF
```

with this line:

```
ddi.dll$(_FLATRELEASEDIR)\SED1355.dllNK SH
```

10. Edit the file MODE.H (located in X:\wince\platform\odo\drivers\display\SED1355) to set the desired screen resolution, color depth (bpp) and panel type. The sample code defaults to a 640x480 color dual passive 16-bit LCD panel. To support one of the other listed panels, change the #define statement.
11. Edit the file PLATFORM.REG to set the same screen resolution and color depth (bpp) as in MODE.H. PLATFORM.REG is located in X:\wince\platform\cepc\files. The display driver section of PLATFORM.REG should be:

```
; Default for EPSON Display Driver
; 640x480 at 8bits/pixel
; Useful Hex Values
; 1024=0x400, 768=0x300 640=0x280 480=0x1E0 320=140 240=0xF0
[HKEY_LOCAL_MACHINE\Drivers\Display\SED1355]
"CxScreen"=dword:280
"CyScreen"=dword:1E0
"Bpp"=dword:8
```

12. Generate the proper building environment by double-clicking on the sample project icon (i.e. X86 DEMO7).
13. Type BLDDemo <ENTER> at the DOS prompt of the X86 DEMO7 window to generate a Windows® CE image file (NK.BIN).

1.3 Example Installation

Installation for Hitachi D9000 and ETMA ODO

Follow the procedures from your Hitachi D9000 (or ETMA ODO) manual and download the following to the D9000 platform:

1. Download SEIKO EPSON's common interface FPGA code (ODO.RBF) to the EEPROM of the D9000 system.
2. Download the Windows® CE binary ROM image (NK.BIN) to the FLASH memory of the D9000 system.

Installation for CEPC Environment

Windows® CE v2.0 can be loaded on a PC using a floppy drive or a hard drive. The two methods are described below:

1. To load CEPC from a floppy drive:
 - a. Create a DOS bootable floppy disk.
 - b. Edit CONFIG.SYS on the floppy disk to contain the following line only.
device=a:\himem.sys
 - c. Edit AUTOEXEC.BAT on the floppy disk to contain the following lines.
mode com1:9600,n,8,1
loadcepc /B:9600 /C:1 /D:2 c:\wince\release\nk.bin
 - d. Copy LOADCEPC.EXE from c:\wince\public\common\oak\bin to the bootable floppy disk.
 - e. Confirm that NK.BIN is located in c:\wince\release.
 - f. Reboot the system from the bootable floppy disk.
2. To load CEPC from a hard drive:
 - a. Copy LOADCEPC.EXE to the root directory of the hard drive.
 - b. Edit CONFIG.SYS on the hard drive to contain the following line only.
device=c:\himem.sys
 - c. Edit AUTOEXEC.BAT on the hard drive to contain the following lines.
mode com1:9600,n,8,1
loadcepc /B:9600 /C:1 /D:2 c:\wince\release\nk.bin
 - d. Confirm that NK.BIN is located in c:\wince\release.
 - e. Reboot the system from the hard drive.

1.4 *Comments*

- Some of the D9000 systems may not be able to provide enough current for your LCD panel to operate properly. If this is the case, an external power supply should be connected to the panel.
- The Seiko Epson Common Interface FPGA code assumes the display buffer starts at 0x12200000 and IO starts at 0x12000000. If the display buffer or IO location is modified, the corresponding entries in the file S1D13505.H have to be changed. S1D13505.H is located in X:\wince\platform\odo\drivers\display\S1D13505 (where X: is the drive letter).
- The driver is CPU independent but will require another ODO.RBF file to support other CPUs when running on the Hitachi D9000 or ETMA ODO platform. Please check with Seiko Epson for the latest supported CPU ODO files.
- As the time of this printing, the drivers have been tested on the SH-3 and x86 CPUs and have only been run with version 2.0 of the ETK. We are constantly updating the drivers so please check our website at www.erd.epson.com, or contact your Seiko Epson or Epson Electronics America sales representative.

AMERICA

EPSON ELECTRONICS AMERICA, INC.

- HEADQUARTERS -

150 River Oaks Parkway
San Jose, CA 95134, U.S.A.
Phone: +1-408-922-0200 Fax: +1-408-922-0238

- SALES OFFICES -

West

1960 E. Grand Avenue
El Segundo, CA 90245, U.S.A.
Phone: +1-310-955-5300 Fax: +1-310-955-5400

Central

101 Virginia Street, Suite 290
Crystal Lake, IL 60014, U.S.A.
Phone: +1-815-455-7630 Fax: +1-815-455-7633

Northeast

301 Edgewater Place, Suite 120
Wakefield, MA 01880, U.S.A.
Phone: +1-781-246-3600 Fax: +1-781-246-5443

Southeast

3010 Royal Blvd. South, Suite 170
Alpharetta, GA 30005, U.S.A.
Phone: +1-877-EEA-0020 Fax: +1-770-777-2637

EUROPE

EPSON EUROPE ELECTRONICS GmbH

- HEADQUARTERS -

Riesstrasse 15
80992 Munich, GERMANY
Phone: +49-(0)89-14005-0 Fax: +49-(0)89-14005-110

DÜSSELDORF BRANCH OFFICE

Altstadtstrasse 176
51379 Leverkusen, GERMANY
Phone: +49-(0)2171-5045-0 Fax: +49-(0)2171-5045-10

UK & IRELAND BRANCH OFFICE

Unit 2.4, Doncastle House, Doncastle Road
Bracknell, Berkshire RG12 8PE, ENGLAND
Phone: +44-(0)1344-381700 Fax: +44-(0)1344-381701

FRENCH BRANCH OFFICE

1 Avenue de l' Atlantique, LP 915 Les Conquerants
Z.A. de Courtaboeuf 2, F-91976 Les Ulis Cedex, FRANCE
Phone: +33-(0)1-64862350 Fax: +33-(0)1-64862355

BARCELONA BRANCH OFFICE

Barcelona Design Center

Edificio Testa, Avda. Alcalde Barrils num. 64-68
E-08190 Sant Cugat del Vallès, SPAIN
Phon:+34-93-544-2490 Fax:+34-93-544-2491

Scotland Design Center

Integration House, The Alba Campus
Livingston West Lothian, EH54 7EG, SCOTLAND
Phone: +44-1506-605040 Fax: +44-1506-605041

ASIA

EPSON (CHINA) CO., LTD.

23F, Beijing Silver Tower 2# North RD DongSanHuan
ChaoYang District, Beijing, CHINA
Phone: 64106655 Fax: 64107319

SHANGHAI BRANCH

7F, High-Tech Bldg., 900, Yishan Road,
Shanghai 200233, CHINA
Phone: 86-21-5423-5577 Fax: 86-21-5423-4677

EPSON HONG KONG LTD.

20/F., Harbour Centre, 25 Harbour Road
Wanchai, Hong Kong
Phone: +852-2585-4600 Fax: +852-2827-4346
Telex: 65542 EPSCO HX

EPSON TAIWAN TECHNOLOGY & TRADING LTD.

14F, No. 7, Song Ren Road,
Taipei 110
Phone: 02-8786-6688 Fax: 02-8786-6660

HSINCHU OFFICE

13F-3, No. 295, Kuang-Fu Road, Sec. 2
HsinChu 300
Phone: 03-573-9900 Fax: 03-573-9169

EPSON SINGAPORE PTE., LTD.

No. 1 Temasek Avenue, #36-00
Millenia Tower, SINGAPORE 039192
Phone: +65-6337-7911 Fax: +65-6334-2716

SEIKO EPSON CORPORATION

KOREA OFFICE

50F, KLI 63 Bldg., 60 Yoido-dong
Yongdeungpo-Ku, Seoul, 150-763, KOREA
Phone: 02-784-6027 Fax: 02-767-3677

GUMI OFFICE

6F, Good Morning Securities Bldg.
56 Songjeong-Dong, Gumi-City, 730-090, KOREA
Phone: 054-454-6027 Fax: 054-454-6093

SEIKO EPSON CORPORATION

ELECTRONIC DEVICES MARKETING DIVISION

IC Marketing Department

IC Marketing & Engineering Group

421-8, Hino, Hino-shi, Tokyo 191-8501, JAPAN
Phone: +81-(0)42-587-5816 Fax: +81-(0)42-587-5624

ED International Marketing Department

421-8, Hino, Hino-shi, Tokyo 191-8501, JAPAN
Phone: +81-(0)42-587-5814 Fax: +81-(0)42-587-5117



In pursuit of “**Saving**” **Technology**, Epson electronic devices.
Our lineup of semiconductors, liquid crystal displays and quartz devices
assists in creating the products of our customers’ dreams.
Epson IS energy savings.

SEIKO EPSON CORPORATION
ELECTRONIC DEVICES MARKETING DIVISION

■ EPSON Electronic Devices Website

<http://www.epsondevice.com>